Objectives of this course

- Learn what Suprtool can do for you
- Learn the basic commands, variations, options, and syntax
- Learn advanced techniques
- Apply what you’ve just learned; get hands-on experience
Basic Modules

- Module 1 - Introduction to Suprtool
- Module 2 - Working with databases
- Module 3 - HPEloquence Issues and Introduction
- Module 4 - Working with Disc Files
- Module 5 - Selecting Records
- Module 6 - Working with Suprlink
More On Suprtool

- Module 7 - Exporting Data to the World
- Module 8 - Extract command
- Module 9 - Latest Features
- Module 10 - Programming with Suprtool2
Optional Modules

- Optional - Working with Speed Demon (MPE only)
- Optional - PowerHouse
- Optional - Editing TurboIMAGE Datasets (MPE only)
- Optional - HowMessy (Turbo-Image only)
Course Material and Structure

- Copies of all the Instructor’s slides are in this workbook
- Refer to the notes below for more details
- Key concepts are repeated multiple times
- Hands-on exercises to apply new knowledge
- Take notes, draw pictures and diagrams to reinforce new information
- Course should be interactive, with a focus on applying techniques to your requirements and applications
Your requests drive Suprtool development...

- Requests are logged in a Knowledge Base
- R&D answers the phone and takes support calls
- Support and R&D personnel exchange ideas via e-mail
- There is always a new version in the works
- **Sales 1-604-501-2001**
- **Support 1-800-453-8970** Office hours are from 8 a.m. - 4 p.m. Pacific time Monday - Friday
- E-mail: support@robelle.com
Communicating with you

- Change notice with every release
- User group meetings
- World Wide Web: http://www.robelle.com/
  (or www.suprtool.com)
- support@robelle.com (Direct e-mail to Support)
<table>
<thead>
<tr>
<th>Introduction to Suprtool</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>What is Suprtool?</td>
<td>3</td>
</tr>
<tr>
<td>A picture of how Suprtool works</td>
<td>7</td>
</tr>
<tr>
<td>Running Suprtool</td>
<td>9</td>
</tr>
<tr>
<td>Doing some basic Suprtool tasks</td>
<td>10</td>
</tr>
<tr>
<td>Redoing commands</td>
<td>16</td>
</tr>
<tr>
<td>Getting on-line Help</td>
<td>18</td>
</tr>
<tr>
<td>Entering commands</td>
<td>20</td>
</tr>
<tr>
<td>Vocabulary</td>
<td>24</td>
</tr>
</tbody>
</table>
What is Suprtool?

- It is a software tool for the HP 3000 and HP 9000
- It extracts data quickly
- It does many data processing functions for files and databases: copies, selects, and sorts, reformats, prints
- It links data from several files into one
- It provides FAST serial processing of “flat” files, KSAM files, TurboIMAGE, Oracle, Allbase and Eloquence databases
What is in Suprtool?

Suprtool has six components on MPE and four on HP-UX:

1. Suprtool - main program
2. Suprlink - linking program
3. STExport - exporting program
4. Dbedit - TurboIMAGE editing utility (MPE only)
5. Speed Demon - TurboIMAGE extracting routines (MPE only)
6. Suprtool2 - interface routines

Documentation on web site.
Why use Suprtool?

- It’s speedy
- It has powerful, easy to use command syntax
- It maximises machine resources
- Its simple commands mean FAST programming
- It integrates well with other tools
- It has powerful reformatting functions
Why is Suprtool/iX so much faster?

- Traditional reports gather information in an inefficient way
- Suprtool uses serial scans to retrieve records in the order they are stored on the disk
- Suprtool sorts data more quickly than the system sort
- Suprtool can quickly extract, sort, and merge information from many sources
Some Suprtool Commands

Process

1. Specify source of records
2. Select records
3. Sort the selected records
4. Specify fields for output
5. Other optional steps
6. Define destination of records

Suprtool Commands

- BASE, GET, INPUT, CHAIN, OPEN, SELECT
- IF, TABLE, CHAIN
- SORT, KEY
- EXTRACT
- DELETE, DUPLICATE, TOTAL, CLEAN
- OUTPUT, PUT, LIST, UPDATE, ADD
**What is a task?**

- A task is Suprtool’s basic unit of work
- A task has one source of records and one destination for output records
- A task is executed using the XEQ or EXIT commands
- A task can be canceled or reset by using the RESET or EA commands
- The job you are doing may require multiple tasks
Getting into Suprtool

- To get into Suprtool for MPE, you simply use the RUN command:
  :run suprtool.pub.robelle
- To get into Suprtool for HPUX you simply type:
  /opt/robelle/bin/suprtool
Copying an entire dataset to a file

- Use the BASE command to access a database and copy dataset to a file
  
  ```
  >base store,5,reader
  >get d-sales
  >output salesout
  >xeq
  ```

- By default, Suprtool creates a new output file
Copying a subset of records to a file

- Use the IF command to select records from a dataset

```plaintext
>get d-inventory
>if on-hand-qty < 5
>list standard
>output testfile
>xeq
```
Looking at the contents of a file

- You can look at any file using INPUT and OUTPUT
- Beware of unprintable characters

```plaintext
> input lowstock
> output *
> xeq
159,19970828,1,50532001,5053,94.49
133,19971016,1,50522001,5052,80.59
138,19971016,4,50522501,5052,41.53
107,19970812,2,50512001,5051,146.39
111,19970916,3,50513001,5051,128.99
IN=5, OUT=5. CPU-Sec=1. Wall-Sec=1.
```
Exercise
Copying the m-customer dataset

- Open the Store database and copy the m-customer dataset into a file called Custfile:
  > base store,1,WRITER
  > get m-customer
  > output custfile
  > xeq

- Then look at the contents of Custfile
  > input custfile
  > list
  > xeq

- Repeat, but create a “link” file:
  > output custfile,link
First Rule of output:

“Unless you have a really good reason not to, always make your output files self-describing”

> output myfile, link
Getting out

There are 4 ways to complete a task:

- XEQ - executes task, remains in Suprtool
- EXIT- executes task, exits Suprtool, suspending if possible
- EXIT ABORT (EA) - cancels task, terminates and exits Suprtool
- EXIT SUSPEND (ES) - puts task “on hold”, suspends and exits Suprtool
Redoing a task

- A task can be easily corrected and repeated if a mistake has been made

- Use these commands to avoid retyping long lists of commands:
  
  LISTREDO
  REDO
  DO
  BEFORE
  MPEX abbreviations

  >do if
  >if on-hand-qty < 5
  >
Do, Redo, and Listredo

- **DO** re-executes the last command or any prior command, as-is
- **REDO** re-executes the last command or any prior command after making changes to the command
- **LISTREDO**
  - List some prior commands, to the screen or to a file
  - Useful for saving work to a file that may become a script

```
> do
> do 5/10
> redo in
> listredo all;unn;out=savefile
```
Getting on-line Help

- Try these Help keywords to access the user manual:
  - Help
  - Help Intro
  - Help News
  - HQ
  - HQ List

- Press “+” to show the Help tree, which lists Help keywords hierarchically
Command conventions

- No command line may be over 256 characters
- Separate multiple commands on the same line with semi-colon ;
- Continue a long line to the next line by ending it with ampersand &
- Append comments to commands using braces {comment}

> base store,5,reader  {read access only}
> get d-sales; item deliv-date,date,yyyyymmdd
> if deliv-date < $today(-30) and product-no = 123456,&
> >>234567,345678
Execute sequences of commands - Use

- Save a set of commands in a file
- Use the file to execute all the commands
- Create usefiles of DEFINE and ITEM commands for datasets and flat files
- Create usefiles with LISTREDO
- Suppress listing the commands with USEQ
- Suprmgr.Pub.Sys or /opt/robelle/suprmgr are always used at startup
- Use files can be “nested”
Set and Verify options

- Enable or disable processing options using SET
- Check the current state of affairs using VERIFY
- Put SET commands for all tasks in Suprmgr files
If Suprtool and the OS do not have the same command name, a leading colon is optional with OS commands. For example,

\[
\begin{align*}
&>:\text{showtime} \quad \text{is equivalent to} \quad >\text{showtime} \quad \{\text{only MPE}\} \\
&>:\text{reset} \quad \text{is not the same as} \quad >\text{reset} \quad \{\text{MPE and Suprtool}\}
\end{align*}
\]

OS commands cannot be abbreviated.

On MPE, you can execute :Run, Command Files, and User Defined Commands (UDC) inside Suprtool.

No more OS commands can be executed in Suprtool after the SET LIMITS MPE OFF command.
Run Suprtool on MPE

- Parm=4; Info="use foo.defs"
  Execute Info string once at startup
  UX: -c"use foo.defs"

- Parm=8; Info="use doit"
  Execute Info string upon each re-activation

- Parm=16
  Copy the input file to the output file

- Parm=32
  Terminate completely; don’t suspend

- Parm=64
  Check with user before exiting
  UX: suprtool -v
Running Suprtool for HP-UX

- Options entered in normal HP-UX conventions
- suprtool [-cv -oc]
- -c "use usefile"
- -v {Verify exit}
- -oc { sets .stoutcount }
  - if [ `cat .stoutcount` -ge 10 ]; then
  - echo “More than 10 records found”
  - fi
# Quick Vocabulary

<table>
<thead>
<tr>
<th>Term</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>database</td>
<td>LIST</td>
</tr>
<tr>
<td>dataset</td>
<td>OUTPUT</td>
</tr>
<tr>
<td>EXTRACT</td>
<td>REDO</td>
</tr>
<tr>
<td>EXIT</td>
<td>record</td>
</tr>
<tr>
<td>file</td>
<td>SORT</td>
</tr>
<tr>
<td>GET</td>
<td>task</td>
</tr>
<tr>
<td>IF</td>
<td>XEQ</td>
</tr>
</tbody>
</table>
Summary

- Six Suprtool components on MPE four on HP-UX
- Documentation and helpful web site.
- Fast processing
- Edit data interactively on MPE
- Basic Suprtool tasks
- On-line Help
Inside Module 2

**Working with Databases**

- Choosing input from databases: Page 2
- Reading an entire dataset serially: Page 6
- Determining fields in a dataset: Page 8
- Reading specific data chains: Page 10
- Choosing Get versus Chain: Page 11
- Listing records: Page 13
- Changing Output Field Structure: Page 14
- Creating Basic Reports: Page 16
- Changing data in datasets: Page 22
Accessing data files

These Suprtool commands access TurboIMAGE and Eloquence datasets:

- BASE
- GET
- CHAIN
- FORM
- PUT
- DELETE
- UPDATE
You can use the BASE command to open a database

> base store, 5, READER

The BASE command without parameters closes a database

A database remains open until a BASE, RESET BASE orRESET ALL command is executed

Eloquence Base command is slightly different; see module 3
How to find datasets in a database

- Use the FORM SETS command to display datasets

> base store.demo
Database password[;]? 
> form sets
Database:STORE.DEMO.APPDEV TPI: SUPERDEX(15015d) 4.0.39

<table>
<thead>
<tr>
<th>Sets:</th>
<th>Num</th>
<th>Type</th>
<th>Count</th>
<th>Capacity</th>
<th>Entry</th>
<th>Load Factor</th>
<th>Length</th>
<th>B/F</th>
</tr>
</thead>
<tbody>
<tr>
<td>M-CUSTOMER</td>
<td>1</td>
<td>M</td>
<td>9</td>
<td>211</td>
<td>12</td>
<td>9 %</td>
<td>55</td>
<td>7</td>
</tr>
<tr>
<td>M-PRODUCT</td>
<td>2</td>
<td>M</td>
<td>2</td>
<td>307</td>
<td>13</td>
<td>4 %</td>
<td>24</td>
<td>12</td>
</tr>
<tr>
<td>M-SUPPLIER</td>
<td>3</td>
<td>M</td>
<td>6</td>
<td>211</td>
<td>3</td>
<td>1 %</td>
<td>49</td>
<td>8</td>
</tr>
<tr>
<td>D-INVENTORY</td>
<td>4</td>
<td>D</td>
<td>6</td>
<td>462</td>
<td>13</td>
<td>3 %</td>
<td>15</td>
<td>22</td>
</tr>
<tr>
<td>D-SALES</td>
<td>5</td>
<td>D</td>
<td>8</td>
<td>602</td>
<td>8</td>
<td>1 %</td>
<td>19</td>
<td>14</td>
</tr>
</tbody>
</table>
More about the Form command

- The FORM command without parameters first defaults to the current input dataset. If no input has been specified, then it defaults to FORM SETS.

- All output is written to the Formout file, which can be redirected to a line printer or a disc file. Currently the Formout file is not available on HP-UX.
Datasets as input sources

- The GET command reads a dataset in one of several ways:
  - It can read the entire dataset serially.
  - It can read a subset of dataset records serially.
  - It can read records at a specified interval (e.g., every 5th record). This kind of sampling is useful for test purposes.

- A database must be open before you can use the GET command.
Warnings using Get

- Suprtool checks the dataset entry count before and after processing, and warns you if it has changed.

- Suprtool permits concurrent changes, but warns you when this happens. If you need exclusive access, open the database in mode-4.

- If you repeatedly receive warnings of new entries, use the SET EOFREAD ON command to read to end-of-file. (Must be specified before the GET command!)
Determining fields in a dataset

- Use FORM `setname` to display the fields in a dataset

```
> form m-customer
```

M-CUSTOMER MASTER SET 1

Entry: Offset
CITY X12 1
CREDIT-RATING J2 13
CUST-ACCOUNT Z8 17 <<SearchField>>
CUST-STATUS X2 25
NAME-FIRST X10 27
NAME-LAST X16 37
STATE-CODE X2 53
STREET-ADDRESS 2X25 55
ZIP-CODE X6 105

Capacity: 211 (7) Entries: 20 Bytes: 110
Defining New Fields

- Create new field definitions:
  > define D-STATUS,25,1,CHAR

- ABSOLUTE definition:
  define field,byteposition,length[,type]
  e.g. > define ord-total,20,4,integer

- RELATIVE definition:
  define field,fieldname[(subscript)][[offset]],length[,type]
  e.g. > define branch-no,cust-code[1],2

Relative defines are associated with a record item, so will stay correct if the field sequence changes.
Reading specific data chains

- If you know the key value(s), use the CHAIN command to search a dataset and select records with the specified key

  >chain d-sales,customer = "123456"

  >chain dtrans,partnum = "A123","B654","G999"

  >chain d-sales,customer = slist  {use a table}

- Even when you know the key values, the GET command may select the same records faster than CHAIN can

  >get d-sales; if $lookup(slist, customer)
Get versus Chain command

**GET**
- Serial access
- Any dataset
- All records
- Physical order
- MR NOBUF reads
- Selection by any data fields

**CHAIN**
- Keyed access
- Only keyed datasets
- Only records with key values
- Forward chain pointers
- DBFIND and DBGET mode-5 and -7
- Selection by key field
Exercise 1
Get versus Chain: quick, choose one!

- Your task is to retrieve records from the infamous ord-line detail dataset which contains 2.3 million records of 308 bytes each. The key values to be selected are in a file called Ordfile. These 162,000 ord-num field values will select 261,000 records from the dataset.

- Your mission, Jim, should you decide to accept it, will be to access the records as quickly as possible, using either the GET command or the CHAIN command. The final results must be sorted in ord-num sequence.

- As always, should you fail, the Secretary will disavow all knowledge of your actions.
Listing data from datasets

- Use the LIST command without parameters to list records whose format is known

```
> get m-customer
> list
> xeq

>GET M-CUSTOMER (1) >OUT $NULL (0)
CITY = Edmonton  CREDIT-RATING = 240000
CUST-ACCT = 10005  CUST-STATUS = 30
NAME-FIRST = Terry  NAME-LAST = Coyle
STATE-CODE = AL
```
By default all fields in the input record are copied to the output record.

The EXTRACT command overrides this default.

extract field [(subscript)][=value][,....]
extract field1\field2

Can have multiple EXTRACT commands

Up to 255 extracted fields

Can specify fieldnames, constants, strings

Output record will be assembled with fields in the same sequence as the EXTRACT commands.
Extract example ..... 

> get m-customer
> extract name-first, name-last
> extract " City: "
> extract city
> output *
> xeq

Wayne     Humphreys        City: Vancouver
Elizabeth Welton           City: Coquitlam
William   Kirk             City: Richmond
Jack      Morrison         City: Calgary
James     Young            City: Edmonton
Percy     Ferguson         City: Coquitlam
Walley    Nisbet           City: Surrey

.......
A quick way to produce basic reports

- Use the LIST STANDARD command to produce a report with a predefined format

Feb 03, 1996  Base STORE.DEMO Set M-CUSTOMER  Page 1
CUST-ACCO  CITY  NAME-FIRST  NAME-LAST
10004  Edmonton  Arthur  Rogers
10005  Edmonton  Terry  Coyle
10015  Edmonton  James  Young
10016  Edmonton  Tara  Bamford

Suprtool lets you customize reports

You can modify reports to improve their appearance or functionality by doing the following:

- changing the report title
- changing heading names
- changing the sort key to make the report contents more meaningful
Customizing a report title and column headings

- It is easy to change your report title or column headings

```plaintext
>get m-customer
>if city = "Edmonton"
>sort name-last
>list standard,title "Customers in Edmonton",&
>> heading "Customer Name ",&
>>    "City ",&
>>    "Account"
>ext name-last,name-first,city,cust-account
>xeq
```
MPE/iX third-party indexing

- Requires Omnidex or Superdex indexing software or HP B-tree support (not currently supported in Suprtool/UX)

- CHAIN command can access third-party or IMAGE indexes
  
  \texttt{>chain \texttt{m-customer,name-last = }"A@"}

- FORM command marks IMAGE fields with third-party indexing as "<<TPI>>", and B-trees as "<<Indexed>>"

- VERIFY BASE command displays name and version of indexing software
Form command shows third-party indexes

```
>form m-customer

<table>
<thead>
<tr>
<th>Master</th>
<th>Set#1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Entry:</td>
<td>Offset</td>
</tr>
<tr>
<td>CITY</td>
<td>X12 1</td>
</tr>
<tr>
<td>CREDIT-RATING</td>
<td>J2 13</td>
</tr>
<tr>
<td>CUST-ACCOUNT</td>
<td>Z8 17</td>
</tr>
<tr>
<td>CUST-STATUS</td>
<td>X2 25</td>
</tr>
<tr>
<td>NAME-FIRST</td>
<td>X10 27</td>
</tr>
<tr>
<td>NAME-LAST</td>
<td>X16 37</td>
</tr>
<tr>
<td>STATE-CODE</td>
<td>X2 53</td>
</tr>
<tr>
<td>STREET-ADDRESS</td>
<td>2X25 55</td>
</tr>
<tr>
<td>POSTAL-CODE</td>
<td>X6 105</td>
</tr>
</tbody>
</table>
```

Capacity: 211 Entries: 20 Entry Length: 55 Blocking: 7
Exercise 2
Create a listing of the Alberta customers

Create the following report from the STORE database:

Mar 20, 1996 20:32    Alberta Customers    Page 1

<table>
<thead>
<tr>
<th>Account#</th>
<th>Name</th>
<th>City</th>
</tr>
</thead>
<tbody>
<tr>
<td>10004</td>
<td>Rogers</td>
<td>Edmonton</td>
</tr>
<tr>
<td>10005</td>
<td>Coyle</td>
<td>Edmonton</td>
</tr>
<tr>
<td>10006</td>
<td>Frahm</td>
<td>Calgary</td>
</tr>
<tr>
<td>10007</td>
<td>Tiernan</td>
<td>Calgary</td>
</tr>
<tr>
<td>10015</td>
<td>Young</td>
<td>Edmonton</td>
</tr>
<tr>
<td>10016</td>
<td>Bamford</td>
<td>Edmonton</td>
</tr>
<tr>
<td>10017</td>
<td>Morrison</td>
<td>Calgary</td>
</tr>
<tr>
<td>10018</td>
<td>Johnston</td>
<td>Calgary</td>
</tr>
</tbody>
</table>
Changing data in datasets

- The Put, Delete and Update commands make changes to the contents of a dataset
- You must open the database in mode 1, 2, 3, or 4
- You can disable the Put, Delete, and Update functions via the Set Limits ReadOnly command
  >set limits readonly on
Moving data into datasets

- We recommend this set of commands to perform a major load of a dataset from a file

  > input loadfile
  > set dumponerror on {default}
  > set defer on
  > set ignore on
  > put m-cust,store.pub,3
  > xeq

- Input file record structure must match the destination dataset structure exactly!
What if the data doesn’t match exactly?

- Use EXTRACT commands to construct the output record

- Use DEFINE and EXTRACT to change storage formats:
  > define amount,1,8,display {...in input file}
  > define new-amount,1,4,integer {new field}
  > extract new-amount = amount

- Field will have attributes as defined, and value from input record, so the output record will contain the 4-byte integer value of the 8-byte display field in the input record.
Deleting selected records from the input dataset

- Open the database in mode-1, -3, or -4
- Access the dataset using GET or CHAIN
- Select records to be deleted with IF command
- Delete the selected records using DELETE
- Optional step: copy the deleted records somewhere else (e.g., OUTPUT file, LIST file, PUT to another dataset)

```
>get d-sales
>item purch-date,date,ymmd
>if purch-date < $date(*-1/*/*)
>delete
>output oldsales.data,append
>xeq
```
Using two passes guarantees safety

>get d-sales
>item purch-date, date, yymmdd
>if purch-date < $date(*-1/*/*)
>output oldsales.data, append
>xeq

>get d-sales
>if purch-date < $date(*-1/*/*)
>delete
>output $null
>xeq
**Update selected records with new values**

- Open the database in mode-1, -2, -3, or -4
- Access the dataset using GET or CHAIN
- Select records to be updated using IF
- Enable updating using UPDATE command; use CIUPDATE parameter to update critical fields
- Specify fields and new values using EXTRACT commands

```plaintext
>get d-sales
>item purch-date,date,yyymmd
>if purch-date < $date(*-1/*/*)
>update
>extract purch-status = "OLD"
>x eq
```
Assigning Calculated Values

>get d-sales
>update
>extract sales-total = & (product-price * sales-qty) + sales-tax
>xeq

Update all records from the D-SALES dataset [no]: yes
Warning: Using DBGET for the input records
IN=8, OUT=8. CPU-Sec=1. Wall-Sec=1.
Set Lock to control concurrent dataset access

- **SET LOCK 1**
  - Lock the dataset and unlock it again around every DELETE, PUT, and UPDATE
  - Least contention with other processes, but slowest option for Suprtool

- **SET LOCK 0**
  - Lock the dataset at the beginning of the task and unlock it only at the end
  - Best performance for Suprtool, but locks out other processes for duration of Suprtool run

- **SET LOCK n**
  - Lock dataset on $n$ DELETE, PUT, or UPDATE transactions, then unlock
  - Compromise between SET LOCK 0 and SET LOCK 1
Summary

- Display datasets
- Field names and formats
- Data chains
- List datasets
- Reports (e.g., standard, customized)
- Third-party indexing
- Adding, deleting, and modifying records
- Changing data formats
- Locking options
## Inside Module 3

### Working with Eloquence

<table>
<thead>
<tr>
<th>Topic</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Commands to access Eloquence databases</td>
<td>2</td>
</tr>
<tr>
<td>Opening and closing a database</td>
<td>3</td>
</tr>
<tr>
<td>Eloquence and Base Command</td>
<td>4</td>
</tr>
<tr>
<td>Put Command</td>
<td>5</td>
</tr>
<tr>
<td>Summary</td>
<td>6</td>
</tr>
</tbody>
</table>
Accessing data files

- These Suprtool commands access Eloquence data files:
  - BASE
  - GET
  - CHAIN
  - FORM
  - PUT
  - DELETE
  - UPDATE
Opening and closing a database

- You can use the BASE command to open a database
  
  \texttt{>base store,5,READER}

- The BASE command without parameters closes a database

- A database remains open until a BASE, RESET BASE or RESET ALL command is executed

- Alternate Base and Put command syntax special to Eloquence
Eloquence and Base Command

- Base Command Syntax
  - base [servername][:server/]database,mode,password
  - base myserver:eloqdb/sample,5,reader
  - base :eloqdb/sample,5,reader
  - base :eloqdb/sample
  - base sample
Put Command

- Put Command allows the same syntax
  - put dataset,[servername][:server/][database]
  - put dataset,myserver:eloqdb/sample
  - Put dataset
Summary

- Base Command Eloquence syntax
- Put Command allows the same syntax
Inside Module 4

Selecting Records with Suprtool

- Using the If command: Page 2
- Comparing fields: Page 4
- Selecting by pattern-matching: Page 7
- Let’s do a crossword puzzle: Page 9
- Dates as selection criteria: Page 10
- Selecting on partial fields: Page 18
- Using tables to select records: Page 23
- Using Tables with Keyed reads: Page 32
Selecting records

- You can use the IF command to choose records by selecting ranges of numbers, dates, or multiple criteria:
  - if sales-qty >= 100 and sales-qty < 5000
  - if cust-status = 10,20,30,35

- Only one IF command is permitted per task.

- Suprtool uses short-circuit evaluation. E.g.:
  - if age > 70 and sex = “M”
  - should be faster than:
  - if sex = “M” and age > 70
More options to specify selection criteria

You can also use these words and signs to select records:

- AND, OR and NOT operators
- parentheses: ) or (  
- relational operators: = < > >= <= <>  
- pattern matching: == and ><
Comparing fields

- You can compare one field to another
  \[ \text{if } \text{deliv-date} = \text{purch-date} \]

- You can compare a numeric field to a calculation
  \[ \text{if } \text{sales-total} <> \text{product-price} \times \text{sales-qty} \]

- You can compare a field to a constant
  \[ \text{if } \text{cust-status} = "\text{OK}","\text{DEAC}" \]
Arithmetic If expressions

- Select records based on arithmetic expressions

> if unit-cost * sales-qty > 10000

> if sales-total < sales-qty * product-price + sales-tax

- Use parentheses to keep things clear
Field types and sizes in comparisons

- Byte and character fields can be different sizes, but...
  - comparison is for length of shorter field
  - comparison ignores last bytes of longer field
Selecting records by pattern-matching

Pattern-matching

- Includes or excludes values in specified fields using these operators
  - == selects records that match pattern
  - >> selects records that do not match pattern

- Can be used only on character fields

- Can specify multiple selection criteria

- Can use special characters to define selection criteria
Special characters in pattern-matching

- Use these special characters to match patterns:

  @ represents any *string* of characters
  ? represents one *alphanumeric* character
  # represents one *numeric* character
  ~ represents zero or more *blanks*
  & indicates the next character is *literal*
Exercise 1
Solve a crossword puzzle

- Use Suprtool to solve this crossword puzzle:
  - an 8 letter word
  - meaning “most befuddled or dazed”
  - second letter is an “o”
  - fourth letter is a “z”

- HINT: Suprtool has a spelling checker. Each word in its dictionary is stored as one record.
Identifying a field as a date

- First use the ITEM command to identify a field as a date:
  >item transaction-date, date, mmddyy
  >item date-of-birth, date, phdate
  >item disbursement-date, date, cyyymmd

- Then use the IF command to select records:
  >if transaction-date = $today and & date-of-birth < $date(1950/01/01) & and disbursement-date >= & $date(*+5/*/*)
**$DATE - Supported Date Formats**

1. YYMMDD   MMDDYY   DDMMYY  
   YYYYMMDD / CCYYMMDD  MMDDYYYY  DDMMYYYY  
2. YYMM      
   YYYYMM / CCYYMM   MMYYYY  
3. CCYY      
4. YYYYMMDD  
5. AAMMDD    MMDDAA   DDMMAA 
   AAMM      
6. YYDDD     CCYYDDD  
7. ASK, Calendar, HPCalendar, Oracle, PHDate, SRNChronos
**Dates as selection criteria**

- You can select records by specifying date criteria
  
  > `item purch-date, date, phdate`
  > `if purch-date = $date(98/11/30) {Nov. 30, 1998}`

- You can also select a range of dates (e.g., all of December 1998)
  
  > `if purch-date > $date(98/11/30) and & purch-date < $date(99/01/01)`
  > `if purch-date >= $date(98/12/01) and & purch-date <= $date(98/12/31)`
Choosing records by relative date

- The $TODAY function optionally accepts an argument that indicates the number of days before or after the current day
  
  $item expiry, date, yymmdd
  $if expiry = $today {today}
  $if expiry = $today (-1) {yesterday}
  $if expiry > $today (+14) {more than 2 weeks away}

- Suprtool converts the $DATE function into a constant
  
  $item date-field, date, mmddyy
  $if date-field = $date(*/*-6/*) {six months ago}
  $if date-field = 091898 {if today is Mar. 18, 1999 (constant)}
Dates must collate correctly for > and <

- $DATE gets converted to a constant
- For ddmmyy or mmddyy dates, the constant is in that format
- ddmmyy and mmddyy dates don't sort properly
- Suprtool rejects greater than or less than comparisons for them
- Error: Invalid date format for the comparison
- Use $STDDATE for non-collating dates
Use $STDDATE for non-collating dates

- Turn a non-collating date into CCYYMMDD format:
  >item purch-date,date,mmddyy
  >if $stddate(purch-date) < $today

- Compare dates in two different formats by converting them both to CCYYMMDD format:
  >item purch-date,date,mmddyy
  >item deliv-date,date,ddmmyyyy
  >if $stddate(purch-date) <= $stddate(deliv-date)

- Dates must be valid for $stddate to work:
  >item purch-date,date,mmddyy
  >if not invalid(purch-date) and & $stddate(purch-date) < $today
**Date Arithmetic**

- You can calculate the difference between 2 dates using the **$days** function

- **$days** converts a date to the **juliandays** date format. I.e. the number of days since a base date (4713 BC)

```plaintext
item purch-date,date,YYYYMMDD
item deliv-date,date,YYYYMMDD
if $days(deliv-date) - $days(purch-date) > 5
```

- Invalid dates return value 0 (zero)
Converting days back to dates

- **Juliandays** date format represents days offset from 4713 BC

- Combine **juliandays** with **$stddate** to convert result of **$days** calculations:
  ```
  >....
  >extract latest-delivery = ($days(date-ord) + 7)
  >xeq
  >...
  >item latest-delivery,date,juliandays
  >item deliv-date,date,YYYYMMDD
  >extract deliv-date = $stddate(latest-delivery)
  ```
Verify that dates are valid

- Use $INVALID to select records with invalid dates
  >item purch-date,date,ymmd
  >if $invalid(purch-date)
  >list standard title “Records with bad dates”

- Or use it to deselect invalid dates
  >if not $invalid(purch-date) and &
  purch-date > $date(*-6/*)
Year 2000 dates

- Some selections generate “invalid” date constants, if the date field cannot hold century information and the constant would be in the next century
  >item purch-date, date, yymmdd
  >if purch-date > $date(*+5/*/*)
  Error: Cannot use a date beyond 1999 for this format

- You can override this error condition
  >set date ifyy2000error off

- Or you can use $STDDATE to assume a century
  >set date cutoff 50
  >if $stddate(purch-date) > $date(*+5/*/*)
$\text{truncate, Mod mod and abs functions}$

- $\text{truncate}$ returns “whole number”, i.e. drops decimals
  $\text{truncate}(127.2 / 12) = 10$

- Mod returns the remainder
  $7 \mod 5 = 2$

- $\text{abs}$ returns the absolute value (no sign)
  $\text{abs}(-121) = 121$
Selecting on parts of a number

- You can select any part of a numeric field with the If command

- Use a divide operation to select on the high-order digits
  \[ \text{if } \text{truncate}(\text{ord-date-yyymmdd} / 100) = 9812 \]

- Use MOD to select on the low-order digits
  \[ \text{if } \text{ord-date-yyymmdd} \mod 100 \leq 15 \]

- Use divide and MOD together to select on middle digits
  \[ \text{if } (\text{truncate}(\text{ord-date-yyymmdd} / 100) \mod 100) \leq 02 \]
Calculating day of week

- Juliandays measures offset from a Monday

- Combine $days$ with mod to calculate day-of-week
  
  ```
  >ite ord=date,date,YYYYMMDD
  >ext day = ($days(dt) mod 7)
  ```

  0 = Monday
  1 = Tuesday
  2 = Wednesday
  ......
  6 = Sunday
Comparing sub-fields

- You can select any part of a character field with the IF command

- If we define a street-address field as 2X25, any part of this field can be selected

  > if street-address(2) = "Canada"
  > if street-address(1,7,2) = "10"
  > if street-address(1,13) = "Marine Drive"
Testing byte type fields

- You can test if a byte type field contains alpha, numeric, alphanumeric or special characters
  
  ```
  >if cust-account = numeric
  >if street-address <> alphanumeric
  ```

- You can also check for an ASCII character by specifying its numeric value or control letter
  
  ```
  >define any-char,1,1,byte
  >if any-char = ^13 {if byte is a Return}
  >if any-char = ^G {if byte is a Bell}
  ```
Checking bits within a field

- The IF command can select records based on bit values in a field
  
  \begin{align*}
  &\text{if} \ cust\text{-status}.(3:1) = 1 \\
  &\text{if} \ cust\text{-status}.(3:2) = 0
  \end{align*}

- Bit checking only works for 16-bit words

- Field must be *Integer* or *Logical*
Extending the If command

You can extend the length of an IF command beyond the 256 character limit by using the $READ function

```
>get m-customer
>if $read
    -name-last == "@Kirk@" and
    -state-code = "BC"
    -and
    -cust-account >
    -12
    -//
```

$READ prompts for the next line of the IF expression until it encounters a Return or a double slash (//)
Creating tables as selection criteria

- The TABLE command creates a set of values that can be used as selection criteria:

  \[
  \text{TABLE } \text{tablename, itemname, table-keyword, table-values}
  \]

  \[
  > \text{table } \text{select,transcode, item,"BUY","SELL"}
  \]

  \[
  > \text{table } \text{cust-table,cust-num, file,custfile}
  \]

- The source of input can be an item value or a file

- The TABLE command sorts values as they are loaded into a table
Table characteristics

- Only one key can be put into a table
- Suprtool can handle up to ten tables
- Each table can have up to two gigabytes of data on MPE
- 500 Mbs in total on HP-UX
- Tables are temporary structures that are reset when a task has been completed
- You can hold a table so it is not reset
- Table values are sorted
When would I use a table?

- Instead of listing all the values
  
  \[\text{if } \text{field} = \text{value1}, \text{value2}, \text{value3}\]

- When there are too many values to fit in an IF command

- When the selection values change occasionally

- When the selection is based on the results of a prior task
Loading a table with values from a file

- If the file containing the values is not sorted, specify FILE as the keyword
  
  \texttt{table states,st-code,file,western.data}
  \texttt{if qty\textunderscore ship < qty\textunderscore order and $lookup(states,st-code)}

- If the file is sorted, specify SORTED as the keyword
  
  \texttt{table states,st-code,sorted,western.data}
  \texttt{if qty\textunderscore ship < qty\textunderscore order and $lookup(states,st-code)}

- The field selected from the input file must have exactly the same format as the table
How does the Table command find a field?

- If the input file is self-describing, Suprtool finds the location of the field via the user label.

- If the file is not self-describing, or the named field is not found in the file label, Suprtool loads the requested data from the start of each record.
Inserting items into a table

- You can also use the TABLE command to insert hardcoded values

- Specify ITEM as the table keyword

  >table states, st-code, **item**, "WA", "OR", "CA"
  >table states, st-code, **item**, "WI", "ID", "NE"
  >table states, st-code, **item**, "NM", "AK", "HI"
  >if cust-status = "OK" and $lookup(states, st-code)
Selecting input records that match a value in a table

- Use the $LOOKUP function with the IF command to select records that match a value in a table
  \[ \text{if } \text{\$lookup(cust-table,cust-acct)} \]

- If the $LOOKUP function finds a match, the expression is true

- If there are multiple conditions in the IF expression, the expression is evaluated faster when $LOOKUP is the last condition
  \[ \text{if status = "10" and } \text{\$lookup(cust-table,cust-acct)} \]

- Use NOT to select records which don’t match table values
Lookup and Data

>get  ord-details
>table cust-table, cust-no, file, custlist, data(state-code)
>if   $lookup(cust-table, cust-no, state-code) = state-code
>output orders
>xeq
Saving and deleting tables

- The HOLD option tells Suprtool to save a table after a task has been completed.
  - `table states,st-code,file,western.data`
  - `table parts,part-no,file,partin,hold`

- The RESET TABLE command clears all the tables. You cannot reset individual tables.
  - `reset table`
Can we find all the invoices for BC customers and sort them by customer ID?

- The invoice records are in the sales detail dataset, but state-code is in the customer master record

```plaintext
> get m-customer
> if state-code = "BC"
> extract cust-account
> output bccust
> xeq
> table bc,cust-account,file,bccust
> get d-sales
> if $lookup(bc,cust-account)
> sort cust-account
> list standard
> xeq
```
Selecting records using the Chain command

- Alternately, you can use the CHAIN command to find the required invoices after you have created an output file of British Columbia customers (Bccust):
  > table brit,cust-account,file,bccust
  > chain d-sales,cust-account=brit
  > list standard
  > xeq

- The CHAIN command performs keyed retrievals for the values in the table.

- No SORT command is necessary because the CHAIN command retrieves the records in the same order as they are found in the table.
String Functions and Features

- $TRIM,$RTRIM,$LTRIM
- $UPPER,$LOWER
- + Operator and Target field
Summary

- IF command
- Field comparison
- IF expressions (Boolean operators, parentheses)
- Pattern-matching
- Date fields
- Sub-field comparisons
- $READ function
- Tables
- Selecting from one file based on criteria in another file
## Inside Module 5

### Working with Files

<table>
<thead>
<tr>
<th>Topic</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Copying files</td>
<td>2</td>
</tr>
<tr>
<td>Working with ordinary disc files</td>
<td>3</td>
</tr>
<tr>
<td>Defining fields</td>
<td>4</td>
</tr>
<tr>
<td>New or existing files</td>
<td>7</td>
</tr>
<tr>
<td>Writing records</td>
<td>10</td>
</tr>
<tr>
<td>Listing records</td>
<td>13</td>
</tr>
<tr>
<td>Printing reports</td>
<td>18</td>
</tr>
<tr>
<td>Calculating field totals</td>
<td>20</td>
</tr>
<tr>
<td>Sorting and removing duplicates</td>
<td>21</td>
</tr>
</tbody>
</table>
Copying files

- Copying an MPE file
  - `input file1`
  - `output file2`
  - `xeq`

- Copying an entire dataset to an MPE file
  - `base store`
  - `get m-customer`
  - `output custdump`
  - `exit`
MPE files vs HP-UX files

- File system differences
- Input command requires more information on HP-UX
- Reclen
- Linefeeds
Working with ordinary disc files

- A source of input files can be ordinary disc files, such as MPE, KSAM, tape files or POSIX files

- You can select, extract, and sort these files

- Listing the Custdump file
  
  > input custdump
  > list char
  > xeq
POSIX filespace vs MPE filespace

- MPE filenames have three parts:
  - file.group.account
  - file.group  
    account is assumed
  - file  
    group and account are assumed

- POSIX filenames can have any number of parts:
  - ./file  
    assumed to be in the current directory
  - ../file  
    in the next level ‘up’
  - ./Letters/PersonalStuff/Mom-1999-04-15
  - /ACCOUNT/GROUP/FILE

- Suprtool can use POSIX files anywhere it can use MPE files
- HP-UX files are similar to MPE’s POSIX files
Define the record structure

- Use the DEFINE command to describe the layout of a flat file

```plaintext
> input custdump
> define account,17,8
> define lastname,37,16
> define credit,13,4,int;item credit,decimal,2
> extract account,lastname,credit
> sort account
> if credit > 2000.00
> list standard
> xeq
```

May 06, 1996 22:28 File: CUSTDUMP Page 1

<table>
<thead>
<tr>
<th>ACCOUNT</th>
<th>LASTNAME</th>
<th>CREDIT</th>
</tr>
</thead>
<tbody>
<tr>
<td>00010003</td>
<td>Melander</td>
<td>2500.00</td>
</tr>
<tr>
<td>00010005</td>
<td>Coyle</td>
<td>2400.00</td>
</tr>
</tbody>
</table>
Let Suprtool maintain field names

- Use the INPUT command to tell Suprtool that a file has the same structure as a dataset

```
> input custdump = m-customer
> item credit-rating,decimal,2
> extract cust-account,name-last,credit-rating
> sort cust-account
> if credit-rating > 2000.00
> list standard
> xeq
```

May 06, 1996 22:38       File: CUSTDUMP       Page 1
CUST-ACCO  NAME-LAST     CREDIT-RATING

10003  Melander          2500.00
10005  Coyle             2400.00
To create, or not to create -- that is the option

- Default is to create a new, permanent file
  > output custdump

- TEMP creates a temporary file
  > output foo,temp

- APPEND adds data to an existing file
  > output blabla,append

- ERASE overwrites an existing file
  > output subfile,erase
To squeeze, or not to squeeze, that is the option

- Sometimes output file capacity (limit) is set higher than the number of records (EOF)
- Sometimes the limit is squeezed to the EOF to save disc space
- You control it with SET SQUEEZE ON or OFF
- To reserve space for appending later, use SET SQUEEZE OFF
To squeeze, or not to squeeze, what is the default?

- If you specify Set Squeeze On or Off, Suprtool will do what you say
- If you don’t specify, Suprtool makes up its own mind
- The Output file will be squeezed except in these cases:
  - input is a file, not a dataset
  - output option is Append or Erase
  - output option is Ask or Num, Query
Writing records

- The OUTPUT command determines where your output records go and in what format
  
  `>output customer,num,data`

- Select one of these output formats:
  
  Data (default) - records are identical to input format
  Key - records contain only the sort keys
  Num - records contain 32-bit input record number
More common record formats

- Additional formats of the OUTPUT command:
  - Num,Key
  - Num,Data
  - Query, ancient self-describing
  - Link, improved self-describing
  - Num,Query
  - Ask, ASK report writer
  - ASCII, human-readable
  - Display, computer-readable
  - PRN, import to PC program
Numrecs controls size of output file

- Limit the number of records selected
- Limit the size of the sort scratch files
- Limit the size of the output file if input is a dataset
- Specify the number of records in a tape file
- Use percentage >100 with SET SQUEEZE OFF to create output file bigger than input file. This provides space for appending records.

>numrecs 100
>numrecs 100000
>numrecs 10%
>numrecs 200%
Listing records

- Listing refers to displaying the records in either a dump format or as simple reports
- Use the LIST command to produce formatted listings of selected records
  
  `list octal, char`
  `list decimal, record`
  `list standard`
  `list hex, char, labels`
List format of nonself-describing files

> in catalog.pub.sys(12/12)
> list
> exit

> IN CATALOG.PUB.SYS (12) >OUT $NULL (12)
00000: 030460 020127 071157 067147 020166 067554 10 Wrong vol
00006: 072555 062440 067556 020114 042145 073043 ume on LDev
00014: 056056 020040 040556 067564 064145 071040 . Another
00022: 060566 060551 066141 061154 062440 024131 available (Y
00030: 027516 024477 020040 020040 020040 020040 /N)?
00036: SAME TO: 000043
00044: 030060 030061 031460 030060 00013000
Some List options for reports

- ONEPERLINE
- NONAME
- NOSKIP
- STANDARD format
- DUPLEX printing
- HEADINGS
- NOREC
Listing one field per line

\$get m-customer
\$list oneperline
\$xeq

\$GET M-CUSTOMER (1) \$OUT $NULL (0)
NAME-FIRST = Terry
NAME-LAST = Coyle
STATE-CODE = AL
CUST-STATUS = 30
Preparing program input by combining List options

- Combine LIST options to format input to other programs

  > get m-customer
  > extract name-last, name-first, city, state-code
  > list norec, noskip, noname, oneperline
  > file suprlist=myinput,new;save;dev=disc;
  >   rec=-80,,f,ascii;nocctl
  > xeq

- Run the program with the file as input

  :run dataload.prog;stdin=myinput

  or....

  :run dataload.prog < myinput
The LIST command writes to an output file called Suprlist, which defaults to $stdlist.

Override the default using a file command:
```
:file suprlist;dev=laser155
```

Listing to a LaserJet

SET PCL command indicates page orientation and font type.
Printing mailing labels

- Use the EXTRACT command with LIST ONEPERLINE to produce mailing labels

```
>get m-customer
>extract " " {blank line}
>extract " " {blank line}
>extract customer-name
>extract street-address(1)
>extract street-address(2)
>extract street-address(3)
>extract " " {blank line}
>list oneperline,noname,noskip,norec
>xeq
```
Calculating totals in numeric fields

- The TOTAL command provides an easy way to sum the contents of one or more numeric fields in selected records.

```
>if state-code = "BC"
>total sales=total
```

- By default, the result is printed to $stdlist or can be redirected to another device.

- If you are using the Suprtool2 interface from a programming language, the total amount is returned to the calling program in the workspace.
Suprtool can sort in several ways

- On any field
- On any part of an input record, not just previously defined fields
- According to multiple sort keys (e.g., primary, secondary)
- Ascending or descending order

MPE files require a DEFINE command to define the field or use the KEY command
Working with duplicate records

- DUPLICATE [ NONE | ONLY ] [ RECORD | KEYS [ n ] ]
  - duplicate none record
  - duplicate none keys 1
  - duplicate only record
  - duplicate only keys

- DUPLICATE NONE KEYS [ n ][ COUNT ][ TOTAL field [ field… ] ]
  - duplicate none keys count
  - duplicate none keys total sales-qty sales-value
  - duplicate none keys count total sales-qty sales-value
**Discarding duplicates from the output file**

- Remove duplicates to get a list of unique values or records
- Based on the whole record or the sort key(s)

**Input file - 6 records**

<table>
<thead>
<tr>
<th>Cust-Account</th>
<th>Value</th>
<th>Date</th>
<th>Account</th>
</tr>
</thead>
<tbody>
<tr>
<td>10003</td>
<td>112.07</td>
<td>19931015</td>
<td>505</td>
</tr>
<tr>
<td>10003</td>
<td>166.00</td>
<td>19931015</td>
<td>505</td>
</tr>
<tr>
<td>10003</td>
<td>219.10</td>
<td>19931015</td>
<td>505</td>
</tr>
<tr>
<td>10016</td>
<td>159.42</td>
<td>19931021</td>
<td>505</td>
</tr>
<tr>
<td>10020</td>
<td>224.15</td>
<td>19931001</td>
<td>505</td>
</tr>
<tr>
<td>10020</td>
<td>167.13</td>
<td>19931028</td>
<td>505</td>
</tr>
</tbody>
</table>

**Output file - 3 records**

<table>
<thead>
<tr>
<th>Cust-Account</th>
<th>Value</th>
<th>Date</th>
<th>Account</th>
</tr>
</thead>
<tbody>
<tr>
<td>10003</td>
<td>112.07</td>
<td>19931015</td>
<td>505</td>
</tr>
<tr>
<td>10016</td>
<td>159.42</td>
<td>19931021</td>
<td>505</td>
</tr>
<tr>
<td>10020</td>
<td>224.15</td>
<td>19931001</td>
<td>505</td>
</tr>
</tbody>
</table>

> `sort cust-account`
> `dup none keys`
Saving only the duplicates

- Remove “originals” to get a list of duplicate values or records
- Exact opposite of DUPLICATE NONE

Input file - same 6 records

<table>
<thead>
<tr>
<th>Cust</th>
<th>Amount</th>
<th>Date</th>
<th>Account</th>
</tr>
</thead>
<tbody>
<tr>
<td>10003</td>
<td>112.07</td>
<td>19931015</td>
<td>505</td>
</tr>
<tr>
<td>10003</td>
<td>166.00</td>
<td>19931015</td>
<td>505</td>
</tr>
<tr>
<td>10003</td>
<td>219.10</td>
<td>19931015</td>
<td>505</td>
</tr>
<tr>
<td>10016</td>
<td>159.42</td>
<td>19931021</td>
<td>505</td>
</tr>
<tr>
<td>10020</td>
<td>224.15</td>
<td>19931001</td>
<td>505</td>
</tr>
<tr>
<td>10020</td>
<td>167.13</td>
<td>19931028</td>
<td>505</td>
</tr>
</tbody>
</table>

>sort cust-account
>dup only keys

Output file - the other 3 records

<table>
<thead>
<tr>
<th>Cust</th>
<th>Amount</th>
<th>Date</th>
<th>Account</th>
</tr>
</thead>
<tbody>
<tr>
<td>10003</td>
<td>166.00</td>
<td>19931015</td>
<td>505</td>
</tr>
<tr>
<td>10016</td>
<td>159.42</td>
<td>19931021</td>
<td>505</td>
</tr>
<tr>
<td>10020</td>
<td>167.13</td>
<td>19931028</td>
<td>505</td>
</tr>
</tbody>
</table>
Counting records

- DUPLICATE COUNT can tell you how many records have the same key

**Input file - 6 records**

10003 112.07 19931015
10003 166.00 19931015
10003 219.10 19931015
10016 159.42 19931021
10020 224.15 19931001
10020 167.13 19931028

**Output file - 3 records**

10003 112.07 19931015 3
10016 159.42 19931021 1
10020 224.15 19931001 2

> get d-sales
> sort cust-account
> duplicate none keys count
Totaling records

DUPLICATE TOTAL calculates a field total for all records with the same key.

Input file - 6 records
10003 112.07 19931015
10003 166.00 19931015
10003 219.10 19931015
10016 159.42 19931021
10020 224.15 19931001
10020 167.13 19931028

>get d-sales
>sort cust-account
>duplicate none keys total sales-total

Output file - 3 records
10003 112.07 19931015 497.17
10016 159.42 19931021 159.42
10020 224.15 19931001 391.28
Exercises
Duplicates, Duplicates, Duplicates, Duplicates

- Exercise 1
  Create a list of all the states/provinces in which we have customers

- Exercise 2
  List all the dates on which we made more than one sale

- Bonus Exercise 3
  List all the sales made on the dates in Exercise 2
  HINT: Requires two passes, and the TABLE command
How to check Suprtool results

- Use the :SHOWJCW command to check the Job Control Word (JCW) after a task has been completed.

- On MPE V and MPE/iX, the SUPRTOOLOUTCOUNT JCW contains the number of records written to the output file (up to 65,535 maximum).

- On MPE/iX, the SUPRTOOLFULLCOUNT variable also contains the output count (no limit).

- On HP-UX, -oc option puts count into .stoutcount.
Summary

- Copy a dataset or a file
- Define new fields
- Select a set of records
- Produce listings
- Specify record formats
- Sorting records
- Checking for duplicates
# Inside Module 6

## Working with Suprlink

<table>
<thead>
<tr>
<th>Topic</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accessing Suprlink</td>
<td>4</td>
</tr>
<tr>
<td>A sample scenario</td>
<td>6</td>
</tr>
<tr>
<td>Self-describing files</td>
<td>15</td>
</tr>
<tr>
<td>Linking files</td>
<td>19</td>
</tr>
<tr>
<td>Adding information</td>
<td>32</td>
</tr>
<tr>
<td>Suprlink requirements</td>
<td>41</td>
</tr>
<tr>
<td>Performance guidelines</td>
<td>43</td>
</tr>
</tbody>
</table>
Suprlink expands Suprtool capabilities

Suprlink

- Adds multi-file linking to Suprtool’s remarkable speed
- Works on IMAGE, KSAM, and MPE files
- Merges up to 8 files into one
- Creates one sorted file as input to your report programs

“We love Suprtool’s speed, but couldn’t we have multiple dataset extracts too?”
Suprlink ties your data together

DB

Suprtool

File1

Suprtool

File2

Suprtool

File3

Suprlink

Combined file

Report program
Three ways to access Suprlink

- Use the RUN command to use Suprlink directly
  
  ```
  :run suprlink.pub.robelle
  +input file1
  +link file2
  +output file3
  +exit
  ```

- Use the Suprtool LINK command to start Suprlink
  
  ```
  :run suprtool.pub.robelle
  >link
  +input file1
  +link file2
  +output file3
  +exit
  >
  ```
Three ways to access Suprlink continued

- Use Suprtool’s LINK command to pass commands to Suprlink
  
  :run suprtool.pub.robelle
  >link input file1
  >link link file2
  >link output file3
  >link exit

- On HP-UX run Suprlink directly.
  
  –/opt/robelle/bin/suprlink
I need all invoices over $100 for British Columbia customers, now!

- Step 1: Identify the required data, and their sources

- Step 2: Use Suprtool to select and sort records from each dataset or file, extracting the required fields

- Step 3: Link the extracted files

- Step 4: Produce the report from the linked file
**What should the report look like?**

<table>
<thead>
<tr>
<th>Account#</th>
<th>Name</th>
<th>Purch Date</th>
<th>Amount</th>
<th>Product#</th>
</tr>
</thead>
<tbody>
<tr>
<td>10003</td>
<td>Melander</td>
<td>19931015</td>
<td>112.07</td>
<td>50511501</td>
</tr>
<tr>
<td></td>
<td></td>
<td>19931015</td>
<td>166.00</td>
<td>50512501</td>
</tr>
<tr>
<td></td>
<td></td>
<td>19931015</td>
<td>219.10</td>
<td>50513001</td>
</tr>
<tr>
<td>10020</td>
<td>Nisbet</td>
<td>19931001</td>
<td>224.15</td>
<td>50511501</td>
</tr>
<tr>
<td></td>
<td>Walley</td>
<td>19931028</td>
<td>167.13</td>
<td>50512501</td>
</tr>
</tbody>
</table>
Step 1: Where are the records located?

- Suprtool’s FORM SETS command lists all the sets in a database opened with the BASE command, and describes their attributes.

- Use the FORM dataset command to list field names in a dataset.

- Use COBOL Copylib or Cognos Qschema listings to get the layouts of non-IMAGE files.
What datasets are in the Store database?

:run suprtool.pub.robelle
>base store.demo
>form sets

Database: STORE.DEMO.ROBELLE

<table>
<thead>
<tr>
<th>Sets:</th>
<th>Set</th>
<th>Item</th>
<th>Count</th>
<th>Capacity</th>
<th>Entry</th>
<th>Load</th>
<th>Entry</th>
<th>Count</th>
<th>Capacity</th>
<th>Factor</th>
<th>Length</th>
<th>B/F</th>
</tr>
</thead>
<tbody>
<tr>
<td>M-CUSTOMER</td>
<td>1 M</td>
<td>9</td>
<td>211</td>
<td></td>
<td>20</td>
<td>9 %</td>
<td>55</td>
<td>7</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>M-PRODUCT</td>
<td>2 M</td>
<td>3</td>
<td>307</td>
<td></td>
<td>13</td>
<td>4 %</td>
<td>24</td>
<td>12</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>M-SUPPLIER</td>
<td>3 M</td>
<td>6</td>
<td>211</td>
<td></td>
<td>3</td>
<td>1 %</td>
<td>49</td>
<td>8</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>D-INVENTORY</td>
<td>4 D</td>
<td>6</td>
<td>462</td>
<td></td>
<td>13</td>
<td>3 %</td>
<td>15</td>
<td>22</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>D-SALES</td>
<td>5 D</td>
<td>8</td>
<td>602</td>
<td></td>
<td>8</td>
<td>1 %</td>
<td>19</td>
<td>14</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
What fields are in the m-customer dataset?

> `form m-customer`

Database: STORE.DEMO.ROBELLE

<table>
<thead>
<tr>
<th>M-CUSTOMER</th>
<th>Master</th>
<th>Set# 1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Entry:</td>
<td>Offset</td>
<td></td>
</tr>
<tr>
<td>CITY</td>
<td>X12</td>
<td>1</td>
</tr>
<tr>
<td>CREDIT-RATING</td>
<td>J2</td>
<td>13</td>
</tr>
<tr>
<td>CUST-ACCOUNT</td>
<td>Z8</td>
<td>17</td>
</tr>
<tr>
<td>CUST-STATUS</td>
<td>X2</td>
<td>25</td>
</tr>
<tr>
<td>NAME-FIRST</td>
<td>X10</td>
<td>27</td>
</tr>
<tr>
<td>NAME-LAST</td>
<td>X16</td>
<td>37</td>
</tr>
<tr>
<td>STATE-CODE</td>
<td>X2</td>
<td>53</td>
</tr>
<tr>
<td>STREET-ADDRESS</td>
<td>2X25</td>
<td>55</td>
</tr>
<tr>
<td>POSTAL-CODE</td>
<td>X6</td>
<td>105</td>
</tr>
</tbody>
</table>

<<Search Field>>

Capacity: 211 (7)  Entries: 20  Bytes: 110
What fields are in the d-sales dataset?

>form d-sales

Database: STORE.DEMO.ROBELLE

<table>
<thead>
<tr>
<th>D-SALES</th>
<th>Detail</th>
<th>Set# 5</th>
<th>Entry:</th>
<th>Offset</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>CUST-ACCOUNT</td>
<td>Z8</td>
<td>1</td>
<td>(!M-CUSTOMER)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>DELIV-DATE</td>
<td>J2</td>
<td>9</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PRODUCT-NO</td>
<td>Z8</td>
<td>13</td>
<td>(M-PRODUCT)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>PRODUCT-PRICE</td>
<td>J2</td>
<td>21</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PURCH-DATE</td>
<td>J2</td>
<td>25</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SALES-QTY</td>
<td>J1</td>
<td>29</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SALES-TAX</td>
<td>J2</td>
<td>31</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SALES-TOTAL</td>
<td>J2</td>
<td>35</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Capacity: 602 (14)  Entries: 8  Bytes: 38
Step 2: Extracting and sorting records

- First, we need to read all the customer records of British Columbia customers and extract the cust-account, name-last, and name-first fields.

- Next, we have to read all the records of invoices over $100 and extract the cust-account, product-no, purch-date, and sales-total fields.

- The cust-account field is common to both records, so we will sort both files by this cust-account.
Reading records of British Columbia customers

- Use Suprtool to select and sort British Columbia customers

```plaintext
>get m-customer
>if state-code = "BC"
>sort cust-account
>extract cust-account,name-last,name-first
>output custfile,temp,link
>xeq
```
**List of British Columbia customers**

```plaintext
>input custfile;list standard;xeq

<table>
<thead>
<tr>
<th>CUST-ACCO</th>
<th>NAME-LAST</th>
<th>NAME-FIRST</th>
</tr>
</thead>
<tbody>
<tr>
<td>10001</td>
<td>Hamilton</td>
<td>Darlene</td>
</tr>
<tr>
<td>10002</td>
<td>Lackner</td>
<td>Gordon</td>
</tr>
<tr>
<td>10003</td>
<td>Melander</td>
<td>John</td>
</tr>
<tr>
<td>10008</td>
<td>Sarafin</td>
<td>Thomas</td>
</tr>
<tr>
<td>...</td>
<td></td>
<td></td>
</tr>
<tr>
<td>10020</td>
<td>Nisbet</td>
<td>Walley</td>
</tr>
</tbody>
</table>
```
What is a self-describing file?

- It is a standard MPE disc file
- It has user labels that contain a mini-dictionary describing record structures
- Use the FORM command to see the structure
Suprlink requires self-describing (SD) files

- Suprlink uses self-describing files as input and creates SD files as output

- The LINK option of the Suprtool OUTPUT command specifies a self-describing file
  >output custfile,temp,link

- In our example, Custfile and Tranfile are self-describing files that Suprlink can use as input
Use Suprtool again to select and sort records with invoices greater than $100

```plaintext
> get d-sales
> item sales-total,decimal,2
> if sales-total > 100.00
> sort cust-account
> sort purch-date
> extract cust-account,sales-total,purch-date,product-no
> output tranfile,temp,link
> xeq
```
List of invoices over $100

```
> input tranfile;list standard;xeq

<table>
<thead>
<tr>
<th>CUST-ACCO</th>
<th>SALES-TOTAL</th>
<th>PURCH-DATE</th>
<th>PRODUCT-NO</th>
</tr>
</thead>
<tbody>
<tr>
<td>10003</td>
<td>112.07</td>
<td>19931015</td>
<td>50511501</td>
</tr>
<tr>
<td>10003</td>
<td>166.00</td>
<td>19931015</td>
<td>50512501</td>
</tr>
<tr>
<td>10003</td>
<td>219.10</td>
<td>19931015</td>
<td>50513001</td>
</tr>
<tr>
<td>10016</td>
<td>159.42</td>
<td>19931021</td>
<td>50532001</td>
</tr>
<tr>
<td>10020</td>
<td>224.15</td>
<td>19931001</td>
<td>50511501</td>
</tr>
</tbody>
</table>
...
Step 3: Linking customer and invoice records

- Use Suprlink to merge the extracted records

:run suprlink.pub.robelle
+input tranfile
+link custfile
+output reptfile,temp
+exit
What is the structure of the merged file?

>form reptfile

File: REPTFILE.DATA.SALES     (SD Version B.00.00)

<table>
<thead>
<tr>
<th>Entry</th>
<th>Offset</th>
<th>Limit: 6</th>
<th>EOF: 5</th>
<th>Entry Length: 50</th>
<th>Blocking: 81</th>
</tr>
</thead>
<tbody>
<tr>
<td>CUST-ACCOUNT</td>
<td>Z8</td>
<td>1</td>
<td>&lt;&lt;Sort# 1 &gt;&gt;</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SALES-TOTAL</td>
<td>I2</td>
<td>9</td>
<td>&lt;&lt; .2 &gt;&gt;</td>
<td></td>
<td></td>
</tr>
<tr>
<td>PURCH-DATE</td>
<td>I2</td>
<td>13</td>
<td>&lt;&lt;Sort# 2 &gt;&gt;</td>
<td></td>
<td></td>
</tr>
<tr>
<td>PRODUCT-NO</td>
<td>Z8</td>
<td>17</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>NAME-LAST</td>
<td>X16</td>
<td>25</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>NAME-FIRST</td>
<td>X10</td>
<td>41</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
How does the merged file look?

```plaintext
>input reptfile;list standard;xeq

<table>
<thead>
<tr>
<th>CUST-ACCO</th>
<th>SALES-TOTAL</th>
<th>PURCH-DATE</th>
<th>PRODUCT-N</th>
<th>NAME-LAST</th>
<th>NAME-FIRST</th>
</tr>
</thead>
<tbody>
<tr>
<td>10003</td>
<td>112.07</td>
<td>19931015</td>
<td>50511501</td>
<td>Melander</td>
<td>John</td>
</tr>
<tr>
<td>10003</td>
<td>166.00</td>
<td>19931015</td>
<td>50512501</td>
<td>Melander</td>
<td>John</td>
</tr>
<tr>
<td>10003</td>
<td>219.10</td>
<td>19931015</td>
<td>50513001</td>
<td>Melander</td>
<td>John</td>
</tr>
<tr>
<td>10020</td>
<td>224.15</td>
<td>19931001</td>
<td>50511501</td>
<td>Nisbet</td>
<td>Walley</td>
</tr>
<tr>
<td>10020</td>
<td>167.13</td>
<td>19931028</td>
<td>50512501</td>
<td>Nisbet</td>
<td>Walley</td>
</tr>
</tbody>
</table>

...
How does the link work?

**Input file - 6 records**

<table>
<thead>
<tr>
<th>Record</th>
<th>Amount</th>
<th>Date</th>
<th>Code</th>
</tr>
</thead>
<tbody>
<tr>
<td>10003</td>
<td>112.07</td>
<td>19931015</td>
<td>505</td>
</tr>
<tr>
<td>10003</td>
<td>166.00</td>
<td>19931015</td>
<td>505</td>
</tr>
<tr>
<td>10003</td>
<td>219.10</td>
<td>19931015</td>
<td>505</td>
</tr>
<tr>
<td>10016</td>
<td>159.42</td>
<td>19931021</td>
<td>505</td>
</tr>
<tr>
<td>10020</td>
<td>224.15</td>
<td>19931001</td>
<td>505</td>
</tr>
<tr>
<td>10020</td>
<td>167.13</td>
<td>19931028</td>
<td>505</td>
</tr>
</tbody>
</table>

**Link file - 12 records**

<table>
<thead>
<tr>
<th>Record</th>
<th>Name</th>
<th>Surname</th>
</tr>
</thead>
<tbody>
<tr>
<td>10001</td>
<td>Hamilton</td>
<td>Darlene</td>
</tr>
<tr>
<td>10002</td>
<td>Lackner</td>
<td>Gordon</td>
</tr>
<tr>
<td>10003</td>
<td>Melander</td>
<td>John</td>
</tr>
<tr>
<td>10008</td>
<td>Sarafin</td>
<td>Thomas</td>
</tr>
<tr>
<td>10009</td>
<td>Oxenbury</td>
<td>Gordon</td>
</tr>
<tr>
<td>10010</td>
<td>Humphreys</td>
<td>Wayne</td>
</tr>
<tr>
<td>10011</td>
<td>Kirk</td>
<td>William</td>
</tr>
<tr>
<td>10012</td>
<td>Ferguson</td>
<td>Percy</td>
</tr>
<tr>
<td>10013</td>
<td>Andersen</td>
<td>Colin</td>
</tr>
<tr>
<td>10020</td>
<td>Nisbet</td>
<td>Walley</td>
</tr>
</tbody>
</table>
How the link works

Input file - 6 records
10003  112.07  19931015  505
10003  166.00  19931015  505
10003  219.10  19931015  505
10016  159.42  19931021  505
10020  224.15  19931001  505
10020  167.13  19931028  505

Link file - 12 records
10001  Hamilton   Darlene
10002  Lackner    Gordon
10003  Melander   John
10008  Sarafin    Thomas
10009  Oxenbury  Gordon
10010  Humphreys  Wayne
10011  Kirk       William
10012  Ferguson   Percy
10013  Andersen   Colin
10014  Nisbet     Walley
10015  Nisbet     Walley
10016  Nisbet     Walley

Output file - 5 records
10003  112.07  19931015  50511501  Melander  John
10003  166.00  19931015  50512501  Melander  John
10003  219.10  19931015  50513001  Melander  John
10020  224.15  19931001  50511501  Nisbet    Walley
10020  167.13  19931028  50512501  Nisbet    Walley
What happens if we reverse the linking order?

>link input custfile
>link link tranfile
>link output reptfile,temp
>link xeq

>input reptfile;list standard;xeq

<table>
<thead>
<tr>
<th>CUST-ACCO</th>
<th>NAME-LAST</th>
<th>NAME-FIRST</th>
<th>SALES-TOTAL</th>
<th>PURCH-DATE</th>
<th>PRODUCT-N</th>
</tr>
</thead>
<tbody>
<tr>
<td>10003</td>
<td>Melander</td>
<td>John</td>
<td>112.07</td>
<td>19931015</td>
<td>50511501</td>
</tr>
<tr>
<td>10020</td>
<td>Nisbet</td>
<td>Walley</td>
<td>224.15</td>
<td>19931001</td>
<td>50511501</td>
</tr>
<tr>
<td>...</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Reversing the input and link files

Input file - 12 records
10001 Hamilton Darlene
10002 Lackner Gordon
10003 Melander John
10008 Sarafin Thomas
10009 Oxenbury Gordon
10010 Humphreys Wayne
10011 Kirk William
10012 Ferguson Percy
10013 Andersen Colin

Link file - 6 records
10003  112.07  19931015  505
10003  166.00  19931015  505
10003  219.10  19931015  505
10016  159.42  19931021  505
10020  224.15  19931001  505
10020  167.13  19931028  505

Output file - 2 records
10003  Melander  John  112.07  19931015  50511501
10020  Nisbet  Walley  224.15  19931001  50511501
What if an invoice does not match a customer record?

- By default, Suprlink drops *input* records without a matching record in the link file.
- Specify LINK OPTIONAL to override this default and include unmatched input records.
- LINK OPTIONAL does not include *link* records without a matching record in the input file.
Including unmatched records

> link input tranfile
> link link custfile optional
> link output reptfile,temp
> link xeq
### Including unmatched input records

#### Input file - 6 records
- 10003 112.07 19931015 505
- 10003 166.00 19931015 505
- 10003 219.10 19931015 505
- **10016 159.42 19931021 505**
- 10020 224.15 19931001 505
- 10020 167.13 19931028 505

#### Link file - 12 records
- 10001 Hamilton Darlene
- 10002 Lackner Gordon
- 10003 Melander John
- 10008 Sarafin Thomas
- 10009 Oxenbury Gordon
- 10010 Humphreys Wayne
- 10011 Kirk William
- 10012 Ferguson Percy
- 10013 Andersen Colin
- **10016 159.42 19931021 50511501 Melander John**
- **10003 112.07 19931015 50512501 Melander John**
- **10003 166.00 19931015 50513001 Melander John**

#### Output file - 6 records
- 10003 112.07 19931015 50511501 Melander John
- 10003 166.00 19931015 50512501 Melander John
- 10003 219.10 19931015 50513001 Melander John
- **10016 159.42 19931021 50532001 Nisbet Walley**
- 10020 224.15 19931001 50511501 Nisbet Walley
- 10020 167.13 19931028 50512501 Nisbet Walley
Step 4: Produce the report

- Use your favorite report writer to format the final report, adding headings, titles, and other features
- The report writer has almost no work to do
- Use Suprtool LIST command if the reporting needs are basic
Suprtool can (almost) produce the report

> input reptfile
> extract cust-account, name-last, name-first, purch-date, &
> sales-total, product-no
> list standard, title "BC Sales over $100", &
> heading "Account# Name
> Purch Date Amount Product#"
> xeq

May 12, 1996 10:10  BC Sales over $100  Page 1

<table>
<thead>
<tr>
<th>Account#</th>
<th>Name</th>
<th>Purch Date</th>
<th>Amount</th>
<th>Product#</th>
</tr>
</thead>
<tbody>
<tr>
<td>10003</td>
<td>Melander</td>
<td>19931015</td>
<td>112.07</td>
<td>50511501</td>
</tr>
<tr>
<td>10003</td>
<td>Melander</td>
<td>19931015</td>
<td>166.00</td>
<td>50512501</td>
</tr>
<tr>
<td>10003</td>
<td>Melander</td>
<td>19931015</td>
<td>219.10</td>
<td>50513001</td>
</tr>
<tr>
<td>10020</td>
<td>Nisbet</td>
<td>19931001</td>
<td>224.15</td>
<td>50511501</td>
</tr>
<tr>
<td>10020</td>
<td>Nisbet</td>
<td>19931028</td>
<td>167.13</td>
<td>50512501</td>
</tr>
</tbody>
</table>
Suprlink Exercise 1

- From the Store database, find all the British Columbia supplied products that have inventories less than 20

- You should include the product number, quantity in stock, as well as the supplier's name and number
Can I add more information to the report?

- The boss has asked to see product descriptions on the report

<table>
<thead>
<tr>
<th>Account#</th>
<th>Name</th>
<th>Purch Date</th>
<th>Amount</th>
<th>Product#</th>
<th>Product</th>
</tr>
</thead>
<tbody>
<tr>
<td>10003</td>
<td>Melander John</td>
<td>19931015</td>
<td>112.07</td>
<td>50511501</td>
<td>Drill</td>
</tr>
<tr>
<td></td>
<td></td>
<td>19931015</td>
<td>166.00</td>
<td>50512501</td>
<td>Drill</td>
</tr>
<tr>
<td></td>
<td></td>
<td>19931015</td>
<td>219.10</td>
<td>50513001</td>
<td>Saw</td>
</tr>
<tr>
<td>10020</td>
<td>Nisbet Walley</td>
<td>19931001</td>
<td>224.15</td>
<td>50511501</td>
<td>Saw</td>
</tr>
<tr>
<td></td>
<td></td>
<td>19931028</td>
<td>167.13</td>
<td>50512501</td>
<td>Jigsaw</td>
</tr>
</tbody>
</table>
Which dataset contains product descriptions?

>form sets

Database: STORE.DEMO.ROBELLE

<table>
<thead>
<tr>
<th>Sets:</th>
<th>Set</th>
<th>Type</th>
<th>Count</th>
<th>Capacity</th>
<th>Entry Count</th>
<th>Load Factor</th>
<th>Entry Length</th>
<th>B/F</th>
</tr>
</thead>
<tbody>
<tr>
<td>M-CUSTOMER</td>
<td>1</td>
<td>M</td>
<td>9</td>
<td>211</td>
<td>20</td>
<td>9 %</td>
<td>55</td>
<td>7</td>
</tr>
<tr>
<td>M-PRODUCT</td>
<td>2</td>
<td>M</td>
<td>3</td>
<td>307</td>
<td>13</td>
<td>4 %</td>
<td>24</td>
<td>12</td>
</tr>
<tr>
<td>M-SUPPLIER</td>
<td>3</td>
<td>M</td>
<td>6</td>
<td>211</td>
<td>3</td>
<td>1 %</td>
<td>49</td>
<td>8</td>
</tr>
<tr>
<td>D-INVENTORY</td>
<td>4</td>
<td>D</td>
<td>6</td>
<td>462</td>
<td>13</td>
<td>3 %</td>
<td>15</td>
<td>22</td>
</tr>
<tr>
<td>D-SALES</td>
<td>5</td>
<td>D</td>
<td>8</td>
<td>602</td>
<td>8</td>
<td>1 %</td>
<td>19</td>
<td>14</td>
</tr>
</tbody>
</table>
What fields are in the product dataset?

>form m-product

Database: STORE.DEMO.ROBELLE

M-PRODUCT Master Set# 2
Entry: Offset
  PRODUCT-DESC X30  1
  PRODUCT-MODEL X10  31
  PRODUCT-NO    Z8   41     <<Search Field>>

Selecting the required fields

- We want to read the **product-no** and **product-desc** fields in the product master dataset

- We want to read **all the fields** in Reptfile

- **Product-no** field is common to both records
Reading product description records

> get m-product
> sort product-no
> extract product-no,product-desc
> output prodfile,temp,link
> xeq
Re-sorting the invoices on the product field

- Suprlink input and link files must have the same sort key, so the invoices have to be re-sorted on the product-no field

```plaintext
>input reptfile
>sort product-no
>output = input
>xeq
```
Linking product descriptions to the invoices

>link input reptfile
>link link prodfile
>link output listfile temp
>link xeq
How does the new report look?

>`input listfile
>`extract cust-account,name-last,name-first,purch-date,sales-total,product-no
>`extract product-desc
>`list standard,title "BC Sales over $100" ,
>`>heading "Account# Name " ,
>`>"Purch Date Amount Product# and Description"
>`sort cust-account
>`sort purch-date
>`xeq

<table>
<thead>
<tr>
<th>Account#</th>
<th>Name</th>
<th>Purch Date</th>
<th>Amount</th>
<th>Product#</th>
<th>and Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>10003</td>
<td>Melander</td>
<td>John</td>
<td>19931015</td>
<td>112.07</td>
<td>50511501 Makita 3/8&quot; Var. Speed Drill</td>
</tr>
<tr>
<td>10003</td>
<td>Melander</td>
<td>John</td>
<td>19931015</td>
<td>166.00</td>
<td>50512501 Makita 8 1/4&quot; Circular Saw</td>
</tr>
<tr>
<td>10003</td>
<td>Melander</td>
<td>John</td>
<td>19931015</td>
<td>219.10</td>
<td>50513001 Makita 1&quot; Jigsaw</td>
</tr>
<tr>
<td>10020</td>
<td>Nisbet</td>
<td>Walley</td>
<td>19931001</td>
<td>224.15</td>
<td>50511501 Makita 3/8&quot; Var. Speed Drill</td>
</tr>
<tr>
<td>10020</td>
<td>Nisbet</td>
<td>Walley</td>
<td>19931028</td>
<td>167.13</td>
<td>50512501 Makita 8 1/4&quot; Circular Saw</td>
</tr>
</tbody>
</table>

IN=5, OUT=5. CPU-Sec=1. Wall-Sec=1.
Suprlink Exercise 2

- Add the product price to the list in Exercise 1 (page 31)

<table>
<thead>
<tr>
<th>SUPPLIER-</th>
<th>PRODUCT-N</th>
<th>ON-HAND-QTY</th>
<th>SUPPLIER-NAME</th>
</tr>
</thead>
<tbody>
<tr>
<td>5051</td>
<td>50512501</td>
<td>7</td>
<td>Makita Canada Inc.</td>
</tr>
<tr>
<td>5051</td>
<td>50511501</td>
<td>5</td>
<td>Makita Canada Inc.</td>
</tr>
<tr>
<td>5051</td>
<td>50512001</td>
<td>2</td>
<td>Makita Canada Inc.</td>
</tr>
<tr>
<td>5051</td>
<td>50513001</td>
<td>3</td>
<td>Makita Canada Inc.</td>
</tr>
<tr>
<td>5052</td>
<td>50521001</td>
<td>10</td>
<td>Black &amp; Decker</td>
</tr>
</tbody>
</table>

...
Specifying Link Fields

- You can specify link fields:
  + input tranfile by cust-account
  + link custfile by account-num

- Useful when files created with ,QUERY instead of ,LINK

- Also useful for specifying a secondary link key:
  + link majors by ssn cmaj

- If field names different in the input file:
  + link majors by ssn cmaj from ssn currmaj
Suprlink requirements

- Suprlink requires enough disc space for the original database, each input file, the final output file, and hidden Sortscr files
- Input and link files must be self-describing files
- Input and link files must be sorted on the same key field
- Link keys can be any type except a floating-point field type
Performance guidelines

- Avoid using Suprlink if repeated sorting is required
- Minimize record sizes by only selecting necessary fields
- Minimize file sizes by only selecting required records
Summary

- Suprlink theory
- Input files versus link files
- Implied record selection
- Optional linking
- Adding more information
- Performance tips
# Exporting Data to the World

<table>
<thead>
<tr>
<th>Topic</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Exporting data to other applications</td>
<td>2</td>
</tr>
<tr>
<td>STExport converts the data</td>
<td>4</td>
</tr>
<tr>
<td>Running STExport</td>
<td>5</td>
</tr>
<tr>
<td>Dates and Decimals</td>
<td>8</td>
</tr>
<tr>
<td>Specifying fieldnames</td>
<td>10</td>
</tr>
<tr>
<td>Creating Web pages</td>
<td>12</td>
</tr>
<tr>
<td>Summary of formatting commands</td>
<td>17</td>
</tr>
<tr>
<td>Resetting defaults</td>
<td>18</td>
</tr>
<tr>
<td>XML command</td>
<td>19</td>
</tr>
<tr>
<td>Clean your data</td>
<td>23</td>
</tr>
</tbody>
</table>
Exporting Data to other Applications

- Extract the data using Suprtool and Suprlink
- Convert the files using STExport
- Transfer the file to the PC
- Import the delimited file
Data needs to be converted

- Image and Eloquence data has:
  - Fixed-width fields
  - Binary storage formats (J2, K2, P28, etc)
  - Structure defined in Root File.

- PC Applications require:
  - Variable-length fields
  - ASCII values for numerics
  - Field delimiters
  - Field name declarations
**STExport converts the data**

- STExport reads self-describing files
- Outputs ASCII files
- Allows you to specify:
  - field delimiters to use
  - date format
  - fieldnames in first record
  - numeric format
  - fixed or variable length
  - quotes on character fields
  - HTML - *table* or *preformatted*
  - XML output
Ways to run STExport

- **On MPE**
  - From the MPE prompt
    
    `:run stexport.pub.robelle`
  - From Suprtool
    
    `>export`
  - From inside of Suprtool
    
    `>export input custsd`
    
    `>export output custexp`
    
    `>export exit`
    
    `In=20. Out=20. CPU-Sec=1. Wall-Sec=1.`

- **On HP-UX**
  
  `/opt/robelle/bin/stexport`
For example ......

> export
$ in custsd
$ out custexp
$ xeq
$ print custexp

"Vancouver",200000,10010,"20","Wayne","Humphreys","BC",.....
"Coquitlam",200000,10014,"20","Elizabeth","Welton","BC",.....
"Richmond",200000,10011,"20","William","Kirk","BC",.....
"Calgary",200000,10017,"20","Jack","Morrison","AL",.....
"Edmonton",200000,10015,"20","James","Young","AL",.....
"Coquitlam",200000,10012,"20","Percy","Ferguson","BC",.....
"Surrey",200000,10020,"20","Walley","Nisbet","BC",.....
In Microsoft Excel

- Transfer to PC, File/Open in Microsoft Excel:
Use Suprtool’s ITEM command to qualify the fields:

```bash
>get d-sales
>item deliv-date,date,YYYYMMDD
>item product-price,decimal,2
>out salesd,link
>x
```

IN=8, OUT=8. CPU-Sec=1. Wall-Sec=1.

```
>form salesd
```

File: SALES.D.HANS.TRAINING (SD Version B.00.00)

<table>
<thead>
<tr>
<th>Entry</th>
<th>Offset</th>
<th>Offset</th>
</tr>
</thead>
<tbody>
<tr>
<td>CUST-ACCOUNT</td>
<td>Z8</td>
<td>1</td>
</tr>
<tr>
<td>DELIV-DATE</td>
<td>I2</td>
<td>9</td>
</tr>
<tr>
<td>&lt;&lt;YYYYMMDD&gt;&gt;</td>
<td></td>
<td></td>
</tr>
<tr>
<td>PRODUCT-NO</td>
<td>Z8</td>
<td>13</td>
</tr>
<tr>
<td>PRODUCT-PRICE</td>
<td>I2</td>
<td>21</td>
</tr>
<tr>
<td>&lt;&lt; .2 &gt;&gt;</td>
<td></td>
<td></td>
</tr>
<tr>
<td>PURCH-DATE</td>
<td>I2</td>
<td>25</td>
</tr>
<tr>
<td>...etc</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Specify date format in STEXPORT:

> export
$ in salesd
$date DDMMYY "/"
$output *
$x

10020, 04/10/97, 50511501, 98.31, 19971000, 2, 2753, 22415
10003, 16/10/97, 50511501, 98.31, 19971016, 1, 1376, 11207
10003, 16/10/97, 50512501, 145.62, 19971016, 1, 2039, 16600
10003, 16/10/97, 50513001, 192.20, 19971016, 1, 2691, 21910
10016, 20/10/97, 50521001, 24.59, 19971020, 3, 1033, 8411
10016, 20/10/97, 50532001, 139.85, 19971020, 1, 1958, 15942
10020, 28/10/97, 50512501, 146.60, 19971028, 1, 2052, 16713
10010, 20/10/97, 50533001, 69.92, 19971020, 1, 979, 7970

In=8. Out=8. CPU-Sec=1. Wall-Sec=1.
Specifying field names

- Use HEADING command to add fieldnames in the first record:
  
  ```plaintext
  $heading "Description","Model"
  $heading add ","Product Code"
  $output *
  $xeq
  "Description","Model","Product Code"
  "Skil 3/8  Variable Speed Drill","#6523",50531501
  "B&D Router","#7613-04",50522001
  "Skil 8 1/2  Circular Saw","#5665",50532501
  .....etc.....
  
  - HEADING FIELDNAMES uses Image field names.
Fixed-length output

- Force fixed-length with COLUMNS command
  
  ```
  $input prodsd
  $columns fixed
  $out *
  $x
  "Description","Model","Product Code"
  "Skil 3/8 Variable Speed Drill","#6523" , 50531501
  "B&D Router" ,"#7613-04" , 50522001
  "Skil Var. Sp. Auto-Scroll Saw" ,"#4560" , 50533001
  "Skil 8 1/2 Circular Saw" ,"#5665" , 50532501
  "B&D Cordless Screwdriver" ,"#9018-04" , 50521001
  ```

- Also see SPACES and ZERO commands
Preparing Data For The Web

- STExport can create HTML files
- Data can be formatted in a table
  - HTML TABLE command
- Or it can be formatted like a List Standard listing
  - HTML PREFORMATTED command
- Formatting is applied by STExport
  - Numeric data is right justified, with decimal points
  - Alpha data is left justified
  - Dates are formatted as you specify
Preparing HTML Tables

- Use the HTML TABLE command
  
  ```
  $input reptfile
  $heading none
  $heading column "Account #"
  $heading column "Amount"
  $heading column "Date"
  $heading column "Product #"
  $heading column "Last Name"
  $heading column "First Name"
  $html table title "Orders" heading "BC Sales over $100"
  $output bcsales
  $xeq
  ```
The table has one column per field, and one row per record.

<table>
<thead>
<tr>
<th>Account #</th>
<th>Amount</th>
<th>Date</th>
<th>Product #</th>
<th>Last Name</th>
<th>First Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>10003</td>
<td>112.07</td>
<td>19951016</td>
<td>50511501</td>
<td>Melander</td>
<td>John</td>
</tr>
<tr>
<td>10003</td>
<td>166.00</td>
<td>19951016</td>
<td>50512501</td>
<td>Melander</td>
<td>John</td>
</tr>
<tr>
<td>10003</td>
<td>219.10</td>
<td>19951016</td>
<td>50513001</td>
<td>Melander</td>
<td>John</td>
</tr>
<tr>
<td>10020</td>
<td>224.15</td>
<td>19951000</td>
<td>50511501</td>
<td>Nisbet</td>
<td>Walley</td>
</tr>
<tr>
<td>10020</td>
<td>167.13</td>
<td>19951028</td>
<td>50512501</td>
<td>Nisbet</td>
<td>Walley</td>
</tr>
</tbody>
</table>
**Listing-style Data**

- Use the PREFORMATTED option instead of TABLE

---

**BC Sales over $100**

<table>
<thead>
<tr>
<th>Account #</th>
<th>Amount</th>
<th>Date</th>
<th>Product #</th>
<th>Last Name</th>
<th>First Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>10003</td>
<td>112.01</td>
<td>19951016</td>
<td>50511501</td>
<td>Melander</td>
<td>John</td>
</tr>
<tr>
<td>10003</td>
<td>166.00</td>
<td>19951016</td>
<td>50512501</td>
<td>Melander</td>
<td>John</td>
</tr>
<tr>
<td>10003</td>
<td>219.10</td>
<td>19951016</td>
<td>50513001</td>
<td>Melander</td>
<td>John</td>
</tr>
<tr>
<td>10020</td>
<td>224.15</td>
<td>19951000</td>
<td>50511501</td>
<td>Nisbet</td>
<td>Walley</td>
</tr>
<tr>
<td>10020</td>
<td>167.13</td>
<td>19951028</td>
<td>50512501</td>
<td>Nisbet</td>
<td>Walley</td>
</tr>
</tbody>
</table>
**HTML Exercise**

Create an HTML Table that looks like this:

**Customer Purchase History**

<table>
<thead>
<tr>
<th>Acct #</th>
<th>Surname</th>
<th>Given Name</th>
<th>Credit Limit</th>
<th>Total Amount Purchased</th>
<th># of Purchases</th>
<th>Earliest Purchase</th>
<th>Latest Purchase</th>
</tr>
</thead>
<tbody>
<tr>
<td>10003</td>
<td>Melander</td>
<td>John</td>
<td>2500.00</td>
<td>497.17</td>
<td>3</td>
<td>10-16-95</td>
<td>10-16-95</td>
</tr>
<tr>
<td>10010</td>
<td>Humphreys</td>
<td>Wayne</td>
<td>2000.00</td>
<td>79.70</td>
<td>1</td>
<td>10-20-95</td>
<td>10-20-95</td>
</tr>
<tr>
<td>10016</td>
<td>Bamford</td>
<td>Tara</td>
<td>2000.00</td>
<td>243.53</td>
<td>2</td>
<td>10-20-95</td>
<td>10-20-95</td>
</tr>
<tr>
<td>10020</td>
<td>Nisbet</td>
<td>Walley</td>
<td>2000.00</td>
<td>391.28</td>
<td>2</td>
<td></td>
<td>10-28-95</td>
</tr>
</tbody>
</table>
## Summary of formatting Commands

<table>
<thead>
<tr>
<th>Command</th>
<th>Options (default underlined)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Columns</td>
<td>Fixed None</td>
</tr>
<tr>
<td>Date</td>
<td>None &lt;format&gt; &lt;&quot;separator&quot;&gt; &lt;invalid &quot; &quot;&gt;</td>
</tr>
<tr>
<td>Decimal</td>
<td>Period Comma</td>
</tr>
<tr>
<td>Delimiter</td>
<td>None Comma Tab &quot;string&quot;</td>
</tr>
<tr>
<td>Floating</td>
<td>Default Fixed Scientific</td>
</tr>
<tr>
<td>Heading</td>
<td>None Fieldnames &quot;string&quot; Column &quot;string&quot;</td>
</tr>
<tr>
<td>HTML</td>
<td>None Preformatted Table</td>
</tr>
<tr>
<td>Quote</td>
<td>None Double Single</td>
</tr>
<tr>
<td>Sign</td>
<td>None Floating Leading Trailing</td>
</tr>
<tr>
<td>Spaces</td>
<td>None Trailing</td>
</tr>
<tr>
<td>Zero</td>
<td>None Leading</td>
</tr>
</tbody>
</table>
Specified settings apply to subsequent tasks

- Suprtool resets most settings at the end of each task
- STExport resets input and output files, but remembers your settings
- Can specify once, and use many times.
- To reset commands, you must set a new preference:
  
  ```
  heading none
  floating default
  delimiter comma
  ...
  ```
XML Command

- XML Output
  - version
  - doctype
  - file
  - record
<?xml version='1.0' encoding='ISO-8859-1'?>
<Orders>
  <Details>
    <CITY>Los Altos</CITY>
    <CREDIT-RATING>100000</CREDIT-RATING>
    <CUST-ACCOUNT>4003302</CUST-ACCOUNT>
    <CUST-STATUS>20</CUST-STATUS>
    <NAME-FIRST>Ralph</NAME-FIRST>
    <NAME-LAST>Perkins</NAME-LAST>
    <STATE-CODE>CA</STATE-CODE>
    <STREET-ADDRESS>Room 655</STREET-ADDRESS>
    <STREET-ADDRESS>Los Altos 040033022</STREET-ADDRESS>
    <ZIP-CODE>93002</ZIP-CODE>
  </Details>
</Orders>
Xml Tag Characters

- Special characters in Tags
- Set xmltagchar “.”
New Stuff

- Escape Command
- SQL import
- Some Database Importers require an “escape” character
- STExport takes care of this for you
Clean your Data

- Clean command
- Replaces certain characters with whatever you choose
- Does all byte fields
Summary

- STExport reformat data for other applications
- Controlling STExport’s output layout
- Can be invoked in 3 ways on MPE
- Creating Web Pages
- Resetting defaults
## Inside Module 8

### Extracting Data

<table>
<thead>
<tr>
<th>Topic</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Using the Extract command</td>
<td>2</td>
</tr>
<tr>
<td>Coercion</td>
<td>3</td>
</tr>
<tr>
<td>$\text{-functions}$</td>
<td>4</td>
</tr>
<tr>
<td>Extract from a table</td>
<td>7</td>
</tr>
</tbody>
</table>
Extract basics

- Extract command “extracts” the data from the input source
- Fields are “placed” in the output file in order of the extract commands
- Extract a range of fields
- Extracting constants
Coercion and Numeric Expressions

- Extract also coerces data from the source to the target
- Changes from one data type to another
  - Define display-field,1,8,display
  - Extract display-field = double-field
- Simple Arithmetic functions
  - Extract total = cost * qty
  - Extract budget99 = actual98 + 1000
  - Extract profit = sales-amt – cost
  - Extract average = total / qty
  - Extract day = ccymmmdd-date mod 100
Date $- functions

- $date
  - specify constant or relative date value in various formats
- $today
  - current system date or calculate a date relative to the system date
- $stddate
  - converts a date in any format to CCYYMMDD
- $days
  - converts the date into a Julian format
String $- functions

- $upper
  - converts all alphabetic characters to uppercase
- $lower
  - converts all alphabetic characters to lowercase
- $trim
  - removes leading and trailing spaces
- $ltrim
  - removes leading spaces
- $rtrim
  - removes trailing spaces
**Numeric $-$ functions**

- **$abs**
  - Returns absolute value of a number
- **$truncate**
  - Returns a number to the left of the decimal place
Extract from a table

- $lookup available in Extract but slightly different
  - Must load the table with data
  - Table mytable,key,file,sdfilename,data(tabledata)
  - File must be self-describing
  - Extract field = $lookup(mytable,key,tabledata)
Extract from a Table Sample

```
>table newprices, prodno, file, bosslist, data(price, desc)
>get part-master
>if $lookup(newprices, prodno)
>update
>extract price = $lookup(newprices, prodno, price)
>extract desc = $lookup(newprices, prodno, desc)
>xeq
```
Summary

- Extract command
- Extract with coercion
- Numeric Expressions
- Extract with $-functions
- Date functions
- String functions
- Numeric functions
- Extract from a table
Inside Module 9

Latest Features in Suprtool

- Variable Substitution
- $Counter
- $Clean and $FindClean
- $Total and $Subtotal
- $SPLIT
- $EDIT
- $NUMBER
- Suprlink’s Join Command
- Output, else
- FastRead (non-MPE Platforms)
- Dbedit (non-MPE Platforms)
- List Command (non-MPE Platforms)
- Dynamic Loading of Eloquence (non-MPE Platforms)
Development is a continuous process

- We are always working on new features
- New version every month or so
Variable Substitution

- MPE version has had Variable Substitution for two years
  - Internal MPE routine
  - Same as used by the CI

- HP-UX now has the same feature
  - Suprtool functions take precedence
  - $stddate, $total, $date, $today are not replaced
  - Variable must be set and exported prior to running program
  - Suprtool command line, $read function
  - STExport and Suprlink as well
  - Must have Set Varsub On
$counter function

- Sequential number function
- Allows you to retain original output order
  - > get morder
  - > def mycount,1,4,double
  - > ext mycount=$counter
  - > ext orderno
  - > out myfile,link
  - > xeq
Clean my data

- $clean function in Suprtool
  - clean “^9”
  - Set cleanchar “ “
  - update
  - extract mybytefield=$clean(mybytefield)
Clean Example

>base mydb,1,,;
>get customer
>clean "^9","^10","^0","^7"
>set cleanchar " "
>update
>ext address(1) = $clean(address(1))
>ext address(2) = $clean(address(2))
>ext address(3) = $clean(address(3))
>xeq
$FindClean Example

- Users did not want to blindly $clean all records
- Some wanted to track strange characters and corruptions
- $FindClean will find fields that could be cleaned

>base members
>get member-file
>Clean special
>If $findclean(name)
>output toclean,link
>xeq
$Total

- $Total and $subtotal functions
- Provide running grand total and subtotals
- Data is stored in a Packed field
$Subtotal$

- Syntax for the $subtotal function in the extract command is:
  - `extract target = $subtotal(field,sort-field)`
- Must specify a sort
- Sort fields must match
  ```
  >def mytot,1,14,packed
  >get orders
  >sort ordnum
  >ext ordnum
  >ext part-number,description,sales-amount
  >ext mytot=$subtotal(sales-amount,ordnum)
  >out sales,link
  >xeq
  ```
$Split

- $split function
- Extracts out variable length strings from data
- Extract from beginning to “/” character
  > extract first-name=$split(name,first,"/”)
  > extract last-name=$split(name,"/",last)
- Can also split on multiple occurrences of a certain character

Consider the following data:

- Armstrong/ Neil/ Patrick
- Green/ Bob/ Miller
- Edwards/ Janine/
- Armstrong/Arthur/Derek
$split details

- Occurrence is honored
- No need to specify an occurrence of one
- Can nest inside a $trim or similar string function
- $split does check for overflow
- First and Last keywords available
$edit

- $edit function
- Converts from numeric or byte to formatted string of bytes
- Uses syntax and rules similar to Cobol Edit-Masks
- Placeholders and Format characters
- Two sets of rules byte type and numeric based on source data type
- >ext formatdate=$edit(a,"xxxx/xx/xx")
- FORMATDATE = 2003/09/24
Numeric source $edit masks

Numeric source $edit masks
>ext a=$edit(int-field,"$$, $$$.99-")
>ext b=$edit(int-field,"99,999.99-")
>ext c=$edit(int-field,"cr99999.99")
>ext d=$edit(int-field,"-$9999.99")
>ext e=$edit(int-field,"**,**.99+")
>ext f=$edit(int-field,"zz,zzz.99+")
>list
>xeq
>IN FILE1SD.NEIL.GREEN (0) >OUT $NULL (0)
A = $11.11-  B = 00,011.11-
C = CR00011.11  D = -$0011.11
E = ****11.11-  F = 11.11
Handling the sign

- +, -, CR and DB allowed
- Depends on state of the data
- Positive, negative, neutral
Rules for $ sign

- Fixed $ sign edit
- Floating $ edit invoked if two "$$" appear in the edit mask
- Suprtool attempts to fixup most odd cases
  - $,123.45 becomes $123.45
Decimal places

- Data is adjusted to number of decimal places
- Default Decimal symbol is “.”
- Can be changed to “,” or any other single character
- Source field decimal value is honored
Currency symbol and overflow

- Currency symbol can be up to four characters
- Set currencysymbol “$”
- Suprtool by default will not stop if overflow occurs
- Set editstoperror on
$number

- $number function
- Converts free-form numbers to numeric in one step
- Honors signs, decimal places and currency symbols
- This means numbers in New-Price can be read by $number:

<table>
<thead>
<tr>
<th>Item-number</th>
<th>New-Price</th>
</tr>
</thead>
<tbody>
<tr>
<td>12345</td>
<td>+123.45</td>
</tr>
<tr>
<td>34563</td>
<td>+ 27.5</td>
</tr>
<tr>
<td>21312</td>
<td>+ 1.545</td>
</tr>
</tbody>
</table>
$number details

- Rounding and decimal places
- Error conditions
- Currency, Decimals and thousand symbols
Suprlink

- Combines two files by common key
- Link command allows for many to one relationship
- Join command allows for many to many relationships
- SQL-like feature
  - Inner Join
  - Outer Join
SQL continued

- Left Outer Join
- Right Outer Join
- Simple Join task
Join continued

- Only one Join per task
- Can specify a secondary key to join on
Output, else

- One Process Output two files
- If condition
- Output, else = if NOT condition
- MPE Temp file
- Other Platforms filename. else
Set FastRead On

- MPE and MR Nobuf
- Set FastRead uses Eloquence Block read routines
- Two to Five times faster
- Less CPU and Wall Time
**Dynamic Loading**

- Dynamically loads whatever version of Eloquence library you have
- Shlib_path
- Most applications providers set this for you
- Enhancement for the future
**Dbedit**

- Edit Single Records
- Popular in MPE version for editing/fixing single records
- Now will work with Eloquence databases on HP-UX
Future

- Reporting?
- More work in a single pass
Inside Module 9

Latest Features in Suprtool

The latest and greatest features

Variable Substitution

$total$
$counter$
$Clean$
We constantly improve and work on our products. Suprtool is no exception. There are always new features being planned, designed, worked on and implemented. This section is to give you up-to-the-minute progress report on some of the latest features in Suprtool.

For Techies

References
**Variable Substitution**

MPE version has had Variable Substitution for two years
- Internal MPE routine
- Same as used by the CI

HP-UX now has the same feature
- Suprtool functions take precedence
- $stddate, $total, $date, $today are not replaced
- Variable must be set and exported prior to running program
- Suprtool command line, $read function
- STExport and Suprlink as well
- Must have Set Varsub On

Suprtool, STExport and Suprlink on MPE have had Variable Substitution for over two years now. We have recently added Variable Substitution for the three programs on HP-UX. The feature is invoked in the same manner, >Set Varsub On. However, you cannot have an environment variable resolved if it has the same name as a Suprtool function, such as $date, $stddate etc.
Suprtool Training Manual

$counter function

Sequential number function
Allows you to retain original output order

> get morder
> def mycount,1,4,double
> ext mycount=$counter
> ext ordeño
> out myfile,link
> xeq

For years, Suprtool has had the ability to output a record number to an output file with the num option of the output command:

> in mysdfile
> out myfile,num,data

The above could would generate an output file called myfile, however, you would lose the SD information and you can only put the number at the beginning or the end of the data. Suprtool now has a counter function that allows you to place a $counter anywhere in the output record as well as preserve the SD information.

> in mysdfile
> def mycount,1,4,double
> ext field1
> ext field2
> ext mycount=$counter
> out myfile,link
> xeq

The file myfile will be self-describing and contain the fields field1, field2 and mycount. The field mycount is defined as a double integer, since this is the only field type that the $counter function can use. Each record will have a unique ascending number starting at one.

For Techies

References

Module 9 – Latest Features
Clean my data

$clean function in Suprtool

- clean "^9"
- Set cleanchar " "
- update
- extract mybytefield=$clean(mybytefield)

The Clean command is used to tell Suprtool which characters it needs to look for in a given byte type field. For example:

```
clean "^9","^10", "."
```

tells Suprtool to replace all occurrences of the tab character (Decimal 9), LineFeed (Decimal 10) and periods to whatever the Clean character is set to.

The Clean command takes both, decimal notation and the character itself. However, it is probably most convenient to use the Decimal notation for the characters that you wish to clean. The Decimal notation is indicated by the caret "^" character. By default, Suprtool replaces any of the characters specified in the clean command with a space. You can change the replacement character with the following set command:

```
>set CleanChar "."
```

This command sets the replacement character to a period. You call the clean function the same way you normally use other functions available to If and Extract. For example:

```
>extract address1=$clean(address1)
```
\textbf{Clean Example}

\begin{verbatim}
>base mydb,1,;
>get customer
>clean "^9","^10","^0","^7"
>set cleanchar " \\
>update
>ext address(1) = $clean(address(1))
>ext address(2) = $clean(address(2))
>ext address(3) = $clean(address(3))
>xeq
\end{verbatim}

The above task will look at the three instances of address and replace all occurrences of the tab, linefeed, null and bell characters with a space.
Suprtool2

Programming with Suprtool2

- Calling Suprtool from a program 2
- Invoking Suprtool for an end-user 3
- Examples of user interfaces for Suprtool functions 4
- Prompting users for selection criteria 8
- Installing the Suprtool2 routine 9
- Three ways COBOL can use Suprtool 11
Calling Suprtool from a program

- There are two ways to execute Suprtool commands for a program:
  1. Run Suprtool first, then run the program
  2. Have the program call Suprtool2
Invoking Suprtool for an end-user

- Any 3GL program can invoke Suprtool tasks, including COBOL, QUICK, FORTRAN, TRANSACT, and SPL

- Call the Suprtool2 interface routine

  ```
  procedure suprtool2 (suprcontrol);
  array suprcontrol;
  ```

- Each call to Suprtool2 passes one line of commands that can include MPE or Suprlink commands

- Suprtool functions are invisible to the end-user
Suprtool2 control Parm

01 supr-control.
  05 supr-version pic s9(4) comp value 4.
  05 supr-status pic s9(4) comp.
    88 supr-ok value zeros.
    88 supr-bad-msgfile value 1.
    88 supr-aborted value 2.
    88 supr-create-error value 3.
    88 supr-bad-total-type value 4.
  05 supr-command-line pic x(256) value spaces.
  05 supr-flags.
    10 supr-priority pic x(2) value spaces.
      88 supr-priority-cs value "CS".
      88 supr-priority-ds value "DS".
      88 supr-priority-es value "ES".
{continued}
Suprtool2 control Parm continued

10 supr-maxdata pic s9(9) comp value 0.
10 supr-print-state pic x(2) value "ER".
  88 supr-print-on-error value "ER".
  88 supr-print-always value "AL".
  88 supr-print-never value "NE".
10 supr-total-type pic x(2) value "CO".
  88 supr-total-cobol value "CO".
  88 supr-total-ascii value "AS".
10 supr-other-flags pic x(18) value spaces.
05 supr-totals pic s9(17) sign is trailing
  separate character occurs 15 times.
05 supr-out-count pic s9(9) comp.
05 supr-workspace pic x(20) value spaces.
$include cobol.qlibsrc.robelle

00-main section.
    perform 02-get-if-specs.
move "base invory.data,5,dev" to supr-command-line.
    perform 01-call-suprtool.
move "get invrec" to supr-command-line.
    perform 01-call-suprtool.
move if-command to supr-command-line.
    perform 01-call-suprtool.
move "purge selitem" to supr-command-line.
    perform 01-call-suprtool.

{continued}
Calls to Suprtool2 from a COBOL program continued

move "output selitem" to supr-command-line.
    perform 01-call-suprtool.
move "extract item,descript" to supr-command-line.
    perform 01-call-suprtool.
move "sort item" to supr-command-line.
    perform 01-call-suprtool.
move "exit" to supr-command-line.
    perform 01-call-suprtool.

Actual call to Suprtool

01-call-suprtool.
call "Suprtool2" using supr-control.
if not supr-ok then
display "Suprtool interface error number: ", supr-status.
Prompting users for selection criteria from a COBOL program

- Use a buffer to format the IF command
  
  01 if-command.
  05 filler   pic x(9) value "IF ITEM=``".
  05 sel-prefix  pic x(4).
  05 filler   pic x(2) value "  ".

- Code to insert selection criteria into the IF buffer
  
  02-get-if-specs.
  display "Enter 4-character item prefix to select:"
  accept input-buf.
  move input-buf to sel-prefix.

- Code in main program
  
  perform 02-get-if-specs.
  move if-command to supr-command-line.
  perform 01-call-suprtool.
Installing Suprtool2 on MPE V

- MPE V uses the CM version of Suprtool2

:run cmprog;lib = p

:segmenter
  -sl sl.pub
  -purgesl segment,suprtool
  -usl st2usl.pub.robelle
  -addsl suprtool
  -exit
Installing Suprtool2 on MPE/iX

- MPE/iX uses the NM version of Suprtool2

- Run your programs with
  
  :run nmprog;xl = "st2xl.pub.robelle"

- Or copy the module to your own XL file

  :linkedit
  -xl xl.pub
  -copy xl;from = st2xl.pub.robelle;& replace
  -exit
Three ways COBOL programs can use Suprtool

- Batch report programs
  - Run Suprtool and create an output file
  - Read and format output file

- On-line programs
  - Call Suprtool2 routine
  - Pass commands to a Suprtool child process
  - Read Suprtool output file

- Call Speed Demon routine instead of DBGET to read every record
Summary

- Call Suprtool2 routine
- Execute Suprtool program as a child process
- Read results from a file created by Suprtool
Editing TurboIMAGE Datasets

- How can Dbedit help me with my work? 3
- Accessing Dbedit 4
- Finding entries with a known key 6
- What if I don't know the key value? 9
- Modifying data and global changes 11
- Dbedit subcommands 14
- Inserting and deleting data 16
- MPE/iX Critical Item Update (CIU) 18
- General guidelines 19
**Editing a database with Dbedit**

- **Dbedit** uses simple commands to perform these editing operations:
  - listing entries
  - adding an entry
  - modifying an entry
  - deleting an entry
  - applying global changes to entries

- It can work on chains of entries or related entries

- It can modify key items
How can Dbedit help me with my work?

- Dbedit is useful in many ways
  - Debugging programs
  - Fixing bad data
  - Building prototype databases
Accessing Dbedit

- Step 1: Run Suprtool

- Step 2: Use the BASE command to open a database

- Step 3: Use the EDIT command to start Dbedit

:run suprtool.pub.robelle
>base store.pub
>edit
#  {Dbedit prompt}
Dbedit is built into Suprtool

- Dbedit is a Suprtool component that functions independently

- Dbedit commands:
  
  #form sets
  #list m-customer
  #modify d-sales;updatekey
  #add d-inventory
  #delete
  #change m-product
  #exit
Finding an entry with a known key

- Use LIST *setname* and specify a key value at the prompt

```
#list m-customer
List in File: M-CUSTOMER
CUST-ACCOUNT >10020
CITY = Surrey
CUST-ACCOUNT = 10020
NAME-FIRST = Walley
STATE-CODE = BC
STREET-ADDRESS = 8877-149th Street
(2)
POSTAL-CODE = V3T4W2

List in File: M-CUSTOMER
CUST-ACCOUNT

Prompts for next value
```
Finding a chain of entries

- Use LIST setname to specify a chain of entries

```
#list d-sales
List in File: D-SALES
   >10020
   >
CUST-ACCOUNT = 10020  PRODUCT-NO >
DELIV-DATE = 19971004
PRODUCT-PRICE = 9831
SALES-QTY = 2
SALES-TOTAL = 22415

CUST-ACCOUNT = 10020  PRODUCT-NO >
DELIV-DATE = 19971028
PRODUCT-PRICE = 14660
SALES-QTY = 1
SALES-TOTAL = 16712
```

List in File: D-SALES
   >
CUST-ACCOUNT >
```
How can I change the search key?

- Use the KEY option to specify a different key and alter the search path

```
#list d-sales;key = product-no
```

List in File: D-SALES

PRODUCT-NO

CUST-ACCOUNT

CUST-ACCOUNT = 10003
PRODUCT-NO = 50512501
PURCH-DATE = 19971016
SALES-TAX = 2039

DELIV-DATE = 19971016
PRODUCT-PRICE = 14562
SALES-QTY = 1
SALES-TOTAL = 16601

CUST-ACCOUNT = 10020
PRODUCT-NO = 50512501
PURCH-DATE = 19971028
SALES-TAX = 2052

DELIV-DATE = 19971028
PRODUCT-PRICE = 14660
SALES-QTY = 1
SALES-TOTAL = 16712
What if I don't know the key value?

- Use the ALL option to sequentially display all the entries in a dataset

```
#list m-customer;all
List ALL records in File: M-CUSTOMER
CITY    = Vancouver    CREDIT-RATING    = 200000
CUST-ACCOUNT = 10010    CUST-STATUS     = 20
NAME-FIRST  = Wayne      NAME-LAST       = Humphreys
STATE-CODE  = BC
STREET-ADDRESS = #403-1075 Comox
             (2)
POSTAL-CODE = V5T1H6

CITY    = Coquitlam    CREDIT-RATING    = 200000
CUST-ACCOUNT = 10014    CUST-STATUS     = 20
NAME-FIRST  = Elizabeth NAME-LAST       = Welton
STATE-CODE  = BC
STREET-ADDRESS = 2788 Oxtoby Place
.....etc....
```
Listing related entries from other datasets

The RELATED option with the LIST command searches for entries in the selected dataset and in related datasets

- If a *master* dataset is specified, Dbedit retrieves a master entry and then goes through the paths to detail sets
  
  \#list m-customer;related

- If a *detail* dataset is specified, Dbedit retrieves a detail chain, then goes through the paths from master sets
  
  \#list d-sales;related
Changing a noncritical field

Use the MODIFY command to change the values of noncritical fields in a record

```
#modify d-inventory : unit-cost
Modify within File: D-INVENTORY
SUPPLIER-NAME > STD Ribbons
PRODUCT-NO > 105391
Enter new values (or <Return> to leave as is):
SUPPLIER-NAME = STD Ribbons
PRODUCT-NO = 105391
UNIT-COST = 500
______ {enter new unit cost}
```
How can I modify a critical field?

- Use the UPDATEKEY option to modify critical items

```
modify d-inventory;updatekey
```

Modify within File: d-inventory

- SUPPLIER-NAME > * ________ {no new value}
- PRODUCT-NO > ________ {press Return to omit}

Enter new values (or <Return> to leave as is):
- SUPPLIER-NAME = STD Ribbons
  
  STD Ribbon {new key value}
Can I make a global change to a field?

- If you need to change a field value in the entire dataset, use the CHANGE command

```
#change m-supplier
Enter existing key value to find:
  SUPPLIER-NAME > ACME
Enter new key value to replace with:
  SUPPLIER-NAME > ACME SUPPLY
SUPPLIER-NAME = ACME  CITY = Los Angeles
STATE-CODE = CA  STREET(1) = 100 Main
STREET(2) =  ZIP-CODE = 91201
OK to change this entry[no]: Y
```
In response to the Dbedit prompt for a field value, you can use the following subcommands:

*  No new value
?  Display the TurboIMAGE format or field
  // Quit the command
  \\ Quit the command
  Ctrl-Y Quit the command
  [ Treat rest of line as data, not as subcommand
  ' Set this field to all blanks (batch use)
  = Execute a calculator command
Moving around in a field list

- Try these subcommands to move to other entries in a field list:
  - `>>` Go to the end of the field list
  - `<<` Go to the beginning of the list
  - `>3` Go three fields forward in the list
  - `<3` Go three fields back in the list
  - `@fieldname` Go to the fieldname
Adding new entries to a dataset

- Use the ADD command to insert a new record into a dataset

```shell
#add m-supplier
Add to File: M-SUPPLIER

SUPPLIER-NAME  > ACME
CITY            > Los Angeles
STATE-CODE      > CA
STREET(1)       > 100 Main
STREET(2)       > ________ {press Return to omit}
ZIP-CODE        > 91201
```
How can I delete an entry?

- It's easy to remove an entry using the DELETE command

```bash
#delete m-supplier
Delete from File: M-SUPPLIER
SUPPLIER-NAME >ACME
SUPPLIER-NAME = ACME CITY = Los Angeles
STATE-CODE = CA STREET(1) = 100 Main
STREET(2) = ZIP-CODE = 91201

Is this the entry to delete[no]: Y
```
MPE/iX Critical Item Update (CIU)

- CIU allows programs to modify critical search and sort fields in detail datasets using DBUPDATE

- By default, IMAGE databases have CIU disabled

- Dbedit requires CIU for the CHANGE command and the UPDATEKEY option with the MODIFY command

- Two ways to enable CIU
  1. `set basename ciupdate = on`
  2. `set basename ciupdate = allowed`
General guidelines

- Dbedit works best on single entries or chains of entries
- Dbedit uses keyed access, but serial access can be specified with the LIST ALL command
- All Dbedit commands support the asterisk (*) subcommand
- All commands support a restrictive field list
- A semicolon (;) separates a command from its options
Summary

- Like a text editor for dataset entries
- ADD, CHANGE, DELETE, LIST, MODIFY
- Updating key values
Exercise

- Open the Store database and copy the m-customer dataset into a file called Custfile

- Then look at the contents of Custfile
Exercise

**GET versus CHAIN: quick, choose one!**

- Ord-Line detail dataset has 2.3 million records of 308 bytes
- Ordfile has 162,000 key values which will select 261,000 records

- `chain`  `ord-line,ord-num=my-table`
  `table`  `my-table,ord-num,file-ordfile`
  `output`  `myfile`
  `xeq`

- `get`  `ord-line`
  `table`  `my-table,ord-num,file,ordfile`
  `if`  `$lookup(my-table,ord-num)`
  `sort`  `ord-num`
  `output`  `myfile`
  `xeq`
Create a listing of the Alberta customers

Create the following report from the STORE database:

<table>
<thead>
<tr>
<th>Account#</th>
<th>Name</th>
<th>City</th>
</tr>
</thead>
<tbody>
<tr>
<td>10004</td>
<td>Rogers</td>
<td>Edmonton</td>
</tr>
<tr>
<td>10005</td>
<td>Coyle</td>
<td>Edmonton</td>
</tr>
<tr>
<td>10006</td>
<td>Frahm</td>
<td>Calgary</td>
</tr>
<tr>
<td>10007</td>
<td>Tiernan</td>
<td>Calgary</td>
</tr>
<tr>
<td>10015</td>
<td>Young</td>
<td>Edmonton</td>
</tr>
<tr>
<td>10016</td>
<td>Bamford</td>
<td>Edmonton</td>
</tr>
<tr>
<td>10017</td>
<td>Morrison</td>
<td>Calgary</td>
</tr>
<tr>
<td>10018</td>
<td>Johnston</td>
<td>Calgary</td>
</tr>
</tbody>
</table>
Exercise
Duplicates, Duplicates, Duplicates, Duplicates

- Exercise 1: Create a list of all the states/provinces in which we have customers.
- Exercise 2: List all the dates on which we made more than one sale.
- Bonus Exercise 3: List all the sales made on those dates. Hints: requires two passes, and the Table command
Suprlink Exercise 1

- From the Store database, find all the products of British Columbia suppliers with inventories less than 20

- You should include the product number, quantity in stock, as well as the supplier's name and number
**Suprlink Exercise 2**

- Add the product price to the list in Exercise 1 (page 31)

<table>
<thead>
<tr>
<th>SUPPLIER-</th>
<th>PRODUCT-N</th>
<th>ON-HAND-QTY</th>
<th>SUPPLIER-NAME</th>
</tr>
</thead>
<tbody>
<tr>
<td>5051</td>
<td>50512501</td>
<td>7</td>
<td>Makita Canada Inc.</td>
</tr>
<tr>
<td>5051</td>
<td>50511501</td>
<td>5</td>
<td>Makita Canada Inc.</td>
</tr>
<tr>
<td>5051</td>
<td>50512001</td>
<td>2</td>
<td>Makita Canada Inc.</td>
</tr>
<tr>
<td>5051</td>
<td>50513001</td>
<td>3</td>
<td>Makita Canada Inc.</td>
</tr>
<tr>
<td>5052</td>
<td>50521001</td>
<td>10</td>
<td>Black &amp; Decker</td>
</tr>
</tbody>
</table>

...
Create an HTML Table that looks like this:

<table>
<thead>
<tr>
<th>Acct #</th>
<th>Surname</th>
<th>Given Name</th>
<th>Credit Limit</th>
<th>Total Amount Purchased</th>
<th># of Purchases</th>
<th>Earliest Purchase</th>
<th>Latest Purchase</th>
</tr>
</thead>
<tbody>
<tr>
<td>10003</td>
<td>Melander</td>
<td>John</td>
<td>2500.00</td>
<td>497.17</td>
<td>3</td>
<td>10-16-95</td>
<td>10-16-95</td>
</tr>
<tr>
<td>10010</td>
<td>Humphreys</td>
<td>Wayne</td>
<td>2000.00</td>
<td>79.70</td>
<td>1</td>
<td>10-20-95</td>
<td>10-20-95</td>
</tr>
<tr>
<td>10016</td>
<td>Bamford</td>
<td>Tara</td>
<td>2000.00</td>
<td>243.53</td>
<td>2</td>
<td>10-20-95</td>
<td>10-20-95</td>
</tr>
<tr>
<td>10020</td>
<td>Nisbet</td>
<td>Walley</td>
<td>2000.00</td>
<td>391.28</td>
<td>2</td>
<td></td>
<td>10-28-95</td>
</tr>
</tbody>
</table>
## HowMessy Exercise #1 (Master)

<table>
<thead>
<tr>
<th>Data Set</th>
<th>Type</th>
<th>Capacity</th>
<th>Entries</th>
<th>Load Factor</th>
<th>Secon-Max daries Blks (Highwater)</th>
<th>Blk Fact</th>
</tr>
</thead>
<tbody>
<tr>
<td>A-MASTER</td>
<td>Ato</td>
<td>14505679</td>
<td>9709758</td>
<td>66.9%</td>
<td>36.8% 2395</td>
<td>29</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Search Field</th>
<th>Max</th>
<th>Ave</th>
<th>Std</th>
<th>Expd Blocks</th>
<th>Avg Blocks</th>
<th>Ineff</th>
<th>Elong-ation</th>
</tr>
</thead>
<tbody>
<tr>
<td>MASTER-KEY</td>
<td>37</td>
<td>1.58</td>
<td>1.26</td>
<td>1.00</td>
<td>1.88</td>
<td>48.5%</td>
<td>1.88</td>
</tr>
</tbody>
</table>
# HowMessy Exercise #2 (Detail)

<table>
<thead>
<tr>
<th>Data Set</th>
<th>Type</th>
<th>Capacity</th>
<th>Entries</th>
<th>Load Factor</th>
<th>Secon-Max daries Blks (Highwater)</th>
<th>Blk Fact</th>
</tr>
</thead>
<tbody>
<tr>
<td>D-ITEMS</td>
<td>Det</td>
<td>620571</td>
<td>119213</td>
<td>19.2%</td>
<td>(242025)</td>
<td>7</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Search Field</th>
<th>Max</th>
<th>Ave</th>
<th>Std</th>
<th>Expd</th>
<th>Avg</th>
<th>Ineff</th>
<th>Elongation</th>
</tr>
</thead>
<tbody>
<tr>
<td>ITEM-NO</td>
<td>3</td>
<td>1.00</td>
<td>0.02</td>
<td>1.00</td>
<td>1.00</td>
<td>0.0%</td>
<td>1.00</td>
</tr>
<tr>
<td>SUPPLIER-NO</td>
<td>23</td>
<td>8.07</td>
<td>3.25</td>
<td>1.77</td>
<td>3.30</td>
<td>28.4%</td>
<td>1.86</td>
</tr>
<tr>
<td>LOCATION</td>
<td>5938</td>
<td>11.62</td>
<td>63.64</td>
<td>2.24</td>
<td>2.53</td>
<td>13.2%</td>
<td>1.13</td>
</tr>
<tr>
<td>BO-STATUS</td>
<td>9999999999.99</td>
<td>0.001703</td>
<td>1.001704</td>
<td>7.00</td>
<td>0.0%</td>
<td>1.00</td>
<td></td>
</tr>
<tr>
<td>DISCOUNT</td>
<td>99999</td>
<td>120.181337.15</td>
<td>3.73</td>
<td>39.37</td>
<td>31.9%</td>
<td>10.55</td>
<td></td>
</tr>
</tbody>
</table>
How messy is your database?

- A database is messy if it takes more I/O than it should
- Unnecessary I/O is still a major limiting factor even on MPE/iX machines
- Databases are messy by nature
- Run HowMessy or DBLOADNG against your database
  - HowMessy is a bonus program for Robelle customers
  - DBLOADNG is a contributed library program
Blocks

- TurboIMAGE does all I/O operations in blocks
- A block may contain many user records
- More entries per block means fewer I/Os
- Fewer I/Os means better performance
Record location in masters

- Search item values must be unique
- Location of entries is determined by a hashing algorithm or a primary address calculation
- Calculation is done on search item value to transform it into a record number between one and the capacity
- Different calculation depending on the search item type
  - X, U, Z, and P give random results
  - I, J, K, R, and E give predictable results
Hashing algorithm

- Customer number AA1000 is transformed into a record number

Customer number AA1000

Record number

25299

Blocking factor = 8

Block 3162

Block 1

Block 3162

Block 12501

Capacity: 100001
Hashing algorithm (no collision)

- Customer number BD2134 gives a different record number in a different block

Blocking factor = 8

Capacity: 100001
Hashing algorithm  
(collision - same block)

- Customer number CL1717 hashes to the same record number as AA1000 location
- TurboIMAGE tries to find an empty location in the same block. If it finds one, no additional I/O is required.
- CL1717 becomes a secondary entry. Primary and secondary entries are linked using pointers that form a chain.

```
  Customer number CL1717
```

```
  Block 3162
  AA1000
  25299
  25302
```
Hashing algorithm (collision - different block)

- Customer number MD4884 collides with AA1000
- No more room in this block. TurboIMAGE reads the following blocks until it finds a free record location.
- In this case, MD4884 is placed two blocks away, which requires two additional I/Os.
An example TurboIMAGE database
### HowMessy sample report

<table>
<thead>
<tr>
<th>Type</th>
<th>Load Factor</th>
<th>Highwater (Blks)</th>
<th>Max Blocks</th>
<th>Avg Blocks</th>
<th>Ineff Ptrs</th>
<th>Elongation</th>
</tr>
</thead>
<tbody>
<tr>
<td>M-Customer Man</td>
<td>71.7%</td>
<td>30.5%</td>
<td>1496</td>
<td>11</td>
<td></td>
<td></td>
</tr>
<tr>
<td>A-Order-No Ato</td>
<td>60.7%</td>
<td>25.7%</td>
<td>1</td>
<td>70</td>
<td></td>
<td></td>
</tr>
<tr>
<td>D-Orders Det</td>
<td>76.9%</td>
<td>(851445)</td>
<td>32</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>D-Ord-Items Det</td>
<td>86.5%</td>
<td>(3470097)</td>
<td>23</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Search Field</th>
<th>Max Chain</th>
<th>Ave Chain</th>
<th>Std Dev</th>
<th>Expd Blocks</th>
<th>Avg Blocks</th>
<th>Ineff Ptrs</th>
<th>Elongation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Customer-No</td>
<td>32</td>
<td>1.92</td>
<td>0.32</td>
<td>1.00</td>
<td>1.90</td>
<td>90.5%</td>
<td>1.90</td>
</tr>
<tr>
<td>Order-No</td>
<td>10</td>
<td>1.35</td>
<td>0.62</td>
<td>1.00</td>
<td>1.00</td>
<td>0.0%</td>
<td>1.00</td>
</tr>
<tr>
<td>!Order-No</td>
<td>1</td>
<td>1.00</td>
<td>0</td>
<td>1.00</td>
<td>1.00</td>
<td>0.0%</td>
<td>1.00</td>
</tr>
<tr>
<td>S Customer-No</td>
<td>80</td>
<td>14.34</td>
<td>17.76</td>
<td>1.75</td>
<td>9.20</td>
<td>57.2%</td>
<td>5.25</td>
</tr>
<tr>
<td>S !Order-No</td>
<td>1604</td>
<td>8.06</td>
<td>35.75</td>
<td>1.36</td>
<td>11.32</td>
<td>72.5%</td>
<td>8.34</td>
</tr>
</tbody>
</table>
# HowMessy sample report (master dataset)

**HowMessy/XL (Version 2.2.1)**

**TurboIMAGE/3000 databases**

**Data Base:** STORE.DATA.INVENT

**By Robelle Consulting Ltd**

**Run on:** MON, JAN 9, 1995, 11:48 AM

**Page:** 1

## Data Set Details

<table>
<thead>
<tr>
<th>Type</th>
<th>Load daries</th>
<th>Blks</th>
<th>Blk</th>
</tr>
</thead>
<tbody>
<tr>
<td>M-Customer</td>
<td>71.7%</td>
<td>178018</td>
<td>11</td>
</tr>
<tr>
<td>A-Order-No</td>
<td>60.7%</td>
<td>768556</td>
<td>70</td>
</tr>
<tr>
<td>D-Orders</td>
<td>76.4%</td>
<td>768558</td>
<td>32</td>
</tr>
<tr>
<td>D-Ord-Items</td>
<td>86.5%</td>
<td>3458511</td>
<td>23</td>
</tr>
</tbody>
</table>

## Search Field Details

<table>
<thead>
<tr>
<th>Field</th>
<th>Max</th>
<th>Ave</th>
<th>Std</th>
<th>Expd Blocks</th>
<th>Avg Blocks</th>
<th>Ineff Ptrs</th>
<th>Elongation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Customer-No</td>
<td>32</td>
<td>1.92</td>
<td>0.32</td>
<td>1.00</td>
<td>1.90</td>
<td>90.5%</td>
<td>1.90</td>
</tr>
<tr>
<td>Order-No</td>
<td>10</td>
<td>1.35</td>
<td>0.62</td>
<td>1.00</td>
<td>1.00</td>
<td>0.0%</td>
<td>1.00</td>
</tr>
<tr>
<td>!Order-No</td>
<td>1</td>
<td>1.00</td>
<td>0</td>
<td>1.00</td>
<td>1.00</td>
<td>0.0%</td>
<td>1.00</td>
</tr>
<tr>
<td>S Customer-No</td>
<td>80</td>
<td>14.34</td>
<td>17.76</td>
<td>1.75</td>
<td>9.20</td>
<td>57.2%</td>
<td>5.25</td>
</tr>
<tr>
<td>S !Order-No</td>
<td>1604</td>
<td>8.06</td>
<td>35.75</td>
<td>1.36</td>
<td>11.32</td>
<td>72.5%</td>
<td>8.34</td>
</tr>
</tbody>
</table>
Interpreting master datasets lines

- Pay attention to the following statistics:
  - High percentage of Secondaries (inefficient hashing)
  - High Maximum Blocks (clustering)
  - High Maximum and Average Chains (inefficient hashing)
  - High Inefficient Pointers (when secondaries exist)
  - High Elongation (when secondaries exist)
Report on m-customer

- The number of Secondaries is not unusually high
- However, there may be problems
  - Records are clustering (high Max Blks)
  - Long synonym chain
  - High percentage of Inefficient Pointers

<table>
<thead>
<tr>
<th>Data Set</th>
<th>Type</th>
<th>Capacity</th>
<th>Entries</th>
<th>Load Factor</th>
<th>Secon- Max Blks (Highwater)</th>
<th>Blk Fact</th>
</tr>
</thead>
<tbody>
<tr>
<td>M-CUSTOMER Man</td>
<td>Man</td>
<td>248113</td>
<td>178018</td>
<td>71.7%</td>
<td>30.5% 1496</td>
<td>11</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Search Field</th>
<th>Max</th>
<th>Ave</th>
<th>Std</th>
<th>Expd</th>
<th>Avg</th>
<th>Ineff</th>
<th>Elong- ation</th>
</tr>
</thead>
<tbody>
<tr>
<td>CUSTOMER-NO</td>
<td>22</td>
<td>1.92</td>
<td>0.32</td>
<td>1.00</td>
<td>1.90</td>
<td>90.5%</td>
<td>1.90</td>
</tr>
</tbody>
</table>
Report on a-order-no

- Very tidy dataset
  - Number of Secondaries is acceptable
  - Max Blks, Ineff Ptrs and Elongation are at the minimum values, even if the maximum chain length is a bit high

<table>
<thead>
<tr>
<th>Type</th>
<th>Data Set</th>
<th>Secon-</th>
<th>Max</th>
<th>Load</th>
<th>daries</th>
<th>Blks</th>
<th>Blk</th>
<th>Factor</th>
<th>(Highwater)</th>
<th>Fact</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>A-ORDER-NO</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Ato</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>1266783</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>768556</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>60.7%</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>25.7%</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>70</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Max</th>
<th>Ave</th>
<th>Std</th>
<th>Expd</th>
<th>Avg</th>
<th>Ineff</th>
<th>Elong-</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ORDER-NO</td>
<td>10</td>
<td>1.35</td>
<td>0.62</td>
<td>1.00</td>
<td>1.00</td>
<td>0.0%</td>
</tr>
<tr>
<td></td>
<td>1.00</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1.00</td>
</tr>
</tbody>
</table>

Search Field:
- ORDER-NO

Chain:
- 10

Chain Dev:
- 1.00

Blocks:
- 1.00

Block Ptrs:
- 0.0%

Elongation:
- 1.00
Master dataset solutions

- Increase capacity to a higher odd number
- Increase the Blocking Factor
  - Increase block size
  - Reduce record size
- Change binary keys to type X, U, Z, or P
- Check your database early in the design
- Use HowMessy on test databases
# HowMessy Exercise 1

<table>
<thead>
<tr>
<th>Data Set</th>
<th>Type</th>
<th>Capacity</th>
<th>Entries</th>
<th>Load Factor</th>
<th>Secon-Max daries Blks (Highwater)</th>
<th>Blk Fact</th>
</tr>
</thead>
<tbody>
<tr>
<td>A-MASTER</td>
<td>Ato</td>
<td>14505679</td>
<td>9709758</td>
<td>66.9%</td>
<td>36.8% 2395</td>
<td>29</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Search Field</th>
<th>Max</th>
<th>Ave</th>
<th>Std</th>
<th>Expd</th>
<th>Avg</th>
<th>Ineff</th>
<th>Elong-</th>
</tr>
</thead>
<tbody>
<tr>
<td>MASTER-KEY</td>
<td>37</td>
<td>1.58</td>
<td>1.26</td>
<td>1.00</td>
<td>1.88</td>
<td>48.5%</td>
<td>1.88</td>
</tr>
</tbody>
</table>
## HowMessy sample report (detail dataset)

<table>
<thead>
<tr>
<th>Type</th>
<th>Data Set</th>
<th>Capacity</th>
<th>Entries</th>
<th>Factor</th>
<th>(Highwater)</th>
<th>Fact</th>
</tr>
</thead>
<tbody>
<tr>
<td>M-CUSTOMER</td>
<td>Man</td>
<td>248113</td>
<td>178018</td>
<td>71.7%</td>
<td>30.5%</td>
<td>1496</td>
</tr>
<tr>
<td>A-ORDER-NO</td>
<td>Ato</td>
<td>126673</td>
<td>768556</td>
<td>60.7%</td>
<td>25.7%</td>
<td>1</td>
</tr>
<tr>
<td>D-ORDERS</td>
<td>Det</td>
<td>1000000</td>
<td>768556</td>
<td>76.9%</td>
<td>851445</td>
<td>12</td>
</tr>
<tr>
<td>D-ORD-ITEMS</td>
<td>Det</td>
<td>4000000</td>
<td>3458511</td>
<td>86.5%</td>
<td>3470097</td>
<td>23</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Search Field</th>
<th>Max</th>
<th>Ave</th>
<th>Std</th>
<th>Expd</th>
<th>Avg</th>
<th>Ineff</th>
<th>Elong-</th>
</tr>
</thead>
<tbody>
<tr>
<td>Customer-No</td>
<td>22</td>
<td>1.92</td>
<td>0.32</td>
<td>1.00</td>
<td>1.90</td>
<td>90.5%</td>
<td>1.90</td>
</tr>
<tr>
<td>Order-No</td>
<td>10</td>
<td>1.35</td>
<td>0.62</td>
<td>1.00</td>
<td>1.00</td>
<td>0.0%</td>
<td>1.00</td>
</tr>
<tr>
<td>!Order-No</td>
<td>1</td>
<td>1.00</td>
<td>0.00</td>
<td>1.00</td>
<td>1.00</td>
<td>0.0%</td>
<td>1.00</td>
</tr>
<tr>
<td>S Customer-No</td>
<td>80</td>
<td>14.34</td>
<td>17.76</td>
<td>1.75</td>
<td>9.20</td>
<td>57.2%</td>
<td>5.25</td>
</tr>
<tr>
<td>S !Order-No</td>
<td>1604</td>
<td>8.06</td>
<td>35.75</td>
<td>1.36</td>
<td>11.32</td>
<td>72.5%</td>
<td>8.34</td>
</tr>
</tbody>
</table>
Empty detail dataset

- Records are stored in the order they are created starting from record 1
- Records for the same customer are linked together using pointers to form a chain
- Chains are linked to the corresponding master entry

Block 1

Block 12500

Capacity: 100000

<table>
<thead>
<tr>
<th>D-ORD-HEADER</th>
<th>Customer</th>
<th>Order</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>AA1000</td>
<td>O000001</td>
</tr>
<tr>
<td>2</td>
<td>MD4884</td>
<td>O000002</td>
</tr>
<tr>
<td>3</td>
<td>BD2134</td>
<td>O000003</td>
</tr>
<tr>
<td>4</td>
<td>MD4884</td>
<td>O000004</td>
</tr>
<tr>
<td>5</td>
<td>CL1717</td>
<td>O000005</td>
</tr>
<tr>
<td>6</td>
<td>AA1000</td>
<td>O000006</td>
</tr>
</tbody>
</table>

Blocking factor = 8
Detail chains get scattered

- Over time, records for the same customer are scattered over multiple blocks
Delete chain

- Deleted records are linked together
- TurboIMAGE reuses the records in the Delete chain, if there are any
Highwater mark

- Indicates highest record location used so far
- Serial reads scan the dataset up to the highwater mark

Used blocks:
- some empty,
- some partially used,
- some full

Diagram:
D-ORD-HEADER

- Block 1
- Block 8000
- Block 12500

highwater mark
Repacking a detail dataset

- Groups records along primary path
- Removes Delete chain (no holes)
- Resets highwater mark
Interpreting detail dataset lines

- Pay attention to the following statistics:
  - Load Factor approaching 100% (dataset full)
  - Primary path (large Average Chain and often accessed)
  - High Average Chain and low Standard deviation, especially with a sorted path (Is path really needed?)
  - High Inefficient Pointers (entries in chain not consecutive)
  - High Elongation (entries in chain not consecutive)
Report on d-orders

- Primary path should be on customer-no, not on order-no
- Highwater mark is high
- Repack along new primary path regularly

<table>
<thead>
<tr>
<th></th>
<th>Type</th>
<th>Capacity</th>
<th>Entries</th>
<th>Load Factor</th>
<th>Secon-Max daries Blks (Highwater)</th>
<th>Blk Fact</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data Set</td>
<td>Det</td>
<td>1000000</td>
<td>768556</td>
<td>76.9%</td>
<td>( 851445)</td>
<td>12</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Search Field</th>
<th>Max</th>
<th>Ave</th>
<th>Std</th>
<th>Expd</th>
<th>Avg</th>
<th>Ineff</th>
<th>Elong-ation</th>
</tr>
</thead>
<tbody>
<tr>
<td>!ORDER-NO</td>
<td>1</td>
<td>1.00</td>
<td>0</td>
<td>1.00</td>
<td>1.00</td>
<td>0.0%</td>
<td>1.00</td>
</tr>
<tr>
<td>S CUSTOMER-NO</td>
<td>80</td>
<td>14.34</td>
<td>17.76</td>
<td>1.75</td>
<td>9.20</td>
<td>57.2%</td>
<td>5.25</td>
</tr>
</tbody>
</table>
Report on d-ord-items

- Inefficient Pointers and Elongation are high
- Highwater mark is fairly high
- Repack the dataset regularly
- Is the sorted path really needed?

<table>
<thead>
<tr>
<th>Data Set</th>
<th>Type</th>
<th>Capacity</th>
<th>Entries</th>
<th>Load Factor</th>
<th>Secon- Max daries Blks</th>
<th>Blk Fact</th>
</tr>
</thead>
<tbody>
<tr>
<td>D-ORD-ITEMS</td>
<td>Det</td>
<td>4000000</td>
<td>3458511</td>
<td>86.5%</td>
<td>(3470097)</td>
<td>23</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Search Field</th>
<th>Max</th>
<th>Ave</th>
<th>Std</th>
<th>Expd</th>
<th>Avg</th>
<th>Ineff</th>
<th>Elong- ation</th>
</tr>
</thead>
<tbody>
<tr>
<td>S !ORDER-NO</td>
<td>1604</td>
<td>8.06</td>
<td>35.75</td>
<td>1.36</td>
<td>11.32</td>
<td>72.5</td>
<td>8.34</td>
</tr>
</tbody>
</table>
Detail dataset solutions

- Assign the primary path correctly; search item with Average Chain length > 1 that is accessed most often
- Repack datasets along the primary path regularly
- Increase the Blocking Factor
  - Increase block size
  - Reduce record size
- Understand sorted paths
- Check your databases early in the design; use HowMessy on test databases
## HowMessy Exercise 2

<table>
<thead>
<tr>
<th>Data Set</th>
<th>Type</th>
<th>Capacity</th>
<th>Entries</th>
<th>Load Factor</th>
<th>Secon-Max Blks (Highwater)</th>
<th>Blk Fact</th>
</tr>
</thead>
<tbody>
<tr>
<td>D-ITEMS</td>
<td>Det</td>
<td>620571</td>
<td>119213</td>
<td>19.2%</td>
<td>( 242025)</td>
<td>7</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Search Field</th>
<th>Max Chain</th>
<th>Ave Chain</th>
<th>Std Dev</th>
<th>Expd Blocks</th>
<th>Avg Blocks</th>
<th>Ineff Ptrs</th>
<th>Elong-ation</th>
</tr>
</thead>
<tbody>
<tr>
<td>S ! ITEM-NO</td>
<td>3</td>
<td>1.00</td>
<td>0.02</td>
<td>1.00</td>
<td>1.00</td>
<td>0.0%</td>
<td>1.00</td>
</tr>
<tr>
<td>S SUPPLIER-NO</td>
<td>23</td>
<td>8.07</td>
<td>3.25</td>
<td>1.77</td>
<td>3.30</td>
<td>28.4%</td>
<td>1.86</td>
</tr>
<tr>
<td>LOCATION</td>
<td>5938</td>
<td>11.62</td>
<td>63.64</td>
<td>2.24</td>
<td>2.53</td>
<td>13.2%</td>
<td>1.13</td>
</tr>
<tr>
<td>BO-STATUS</td>
<td>99999</td>
<td>99999.99</td>
<td>0.00</td>
<td>17031.00</td>
<td>17047.00</td>
<td>14.3%</td>
<td>1.00</td>
</tr>
<tr>
<td>DISCOUNT</td>
<td>99999</td>
<td>120.18</td>
<td>1337.15</td>
<td>3.73</td>
<td>39.37</td>
<td>31.9%</td>
<td>10.55</td>
</tr>
</tbody>
</table>
## Minimum number of disc I/Os

<table>
<thead>
<tr>
<th>Intrinsic</th>
<th>Disc I/Os</th>
</tr>
</thead>
<tbody>
<tr>
<td>DBGET</td>
<td>1</td>
</tr>
<tr>
<td>DBFIND</td>
<td>1</td>
</tr>
<tr>
<td>DBBEGIN</td>
<td>1</td>
</tr>
<tr>
<td>DBEND</td>
<td>1</td>
</tr>
<tr>
<td>DBUPDATE</td>
<td>1 (non-critical item)</td>
</tr>
<tr>
<td>DBUPDATE</td>
<td>13 (critical item)</td>
</tr>
<tr>
<td>DBPUT</td>
<td>3 [+ (4 x #paths, if detail)]</td>
</tr>
<tr>
<td>DBDELETE</td>
<td>2 [+ (4 x #paths, if detail)]</td>
</tr>
</tbody>
</table>

Serial reads:
- Master: Capacity / Blocking factor
- Detail: # entries / Blocking factor
Estimating response time

- Deleting 100,000 records from a detail dataset with two paths would take:
  - $2 + (4 \times 2 \text{ paths}) = 10 \text{ I/Os per record}$
  - $100,000 \text{ records} \times 10 \text{ I/Os per record} = 1,000,000 \text{ I/Os}$
- Classic: around 25 I/Os per second
  - $1,000,000 \text{ I/Os} / 25 = 40,000 \text{ seconds}$
  - $40,000 \text{ seconds} / 3600 = 11.1 \text{ hours}$
- iX: around 40 I/Os per second
  - $1,000,000 \text{ I/Os} / 40 = 25,000 \text{ seconds}$
  - $25,000 \text{ seconds} / 3600 = 6.9 \text{ hours}$
Automating HowMessy analysis

- Recent version of HowMessy creates a self-describing file with these statistics
- Process the file with generic tools (Suprtool, AskPlus) or custom programs (COBOL, 4GL), and produce custom reports
- Send messages to database administrators
- Write “smart” job to fix databases without user intervention
Datasets more than 80% full

```plaintext
>input loadfile
>if loadfactor > 80
>ext database, dataset, datasettype, loadfactor
>list standard
```

Only one address per customer

```plaintext
>input loadfile
>if dataset = "D-ADDRESSES" and &
    maxchain > 1
```
References

- The TurboIMAGE/3000 Handbook (Chapter 23)

- Available for $49.95 from:
  
  WORDWARE
  P.O. Box 14300
  Seattle, WA 98114
Summary

- TurboIMAGE databases become messy over time, especially if they are active.

- HowMessy and DBLOADNG let you analyze the database’s efficiency.

- You should have some knowledge of the internal workings of TurboIMAGE.

- Monitor your databases regularly.
## PowerHouse and Suprtool

<table>
<thead>
<tr>
<th>Feature</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Why use Suprtool?</td>
<td>2</td>
</tr>
<tr>
<td>Combining Suprtool and QUIZ</td>
<td>4</td>
</tr>
<tr>
<td>Linking multiple files</td>
<td>8</td>
</tr>
<tr>
<td>Creating complex subfiles</td>
<td>14</td>
</tr>
<tr>
<td>Importing external data for QUIZ</td>
<td>15</td>
</tr>
<tr>
<td>Suprtool subtotal and count operations</td>
<td>16</td>
</tr>
</tbody>
</table>
Why preselect data with Suprtool?

- Suprtool features
  - Fast and efficient serial reads of files
  - Powerful and flexible selection features
  - Efficient sort routines
  - Links files on any field with minimum disc I/O
  - Interfaces with many application tools

- QUIZ features
  - Powerful, flexible report writer
  - Uses standard data retrieval methods
Reading input files

- Suprtool can read
  - IMAGE datasets
  - KSAM files
  - MPE disc files
  - Tape files
  - Other files with fixed-length records

- QUIZ can read
  - QDD or PDL declared files
  - PowerHouse subfiles

- MPE disc files can be declared in the PowerHouse dictionary
A typical QUIZ and Suprtool task

- Choose an input method for QUIZ; Suprtool cannot create subfiles
  1. Create an empty subfile
  or
  2. Describe a direct or sequential file in PDL or QDD

- Use Suprtool to populate the file

- Access the output file in QUIZ and link to others
Step 1: Creating subfiles

- QUIZ
  - access D-SALES
  - report summary all
  - set subfile name SALESUB keep size 10000
  - set report limit 1
  - go

- QTP
  - access D-SALES
  - subfile SALESUB keep size 10000 include D-SALES
  - set input limit 0
  - go

Different results if items redefined more than once
Step 2: Populate the subfile with Suprtool

:run SUPRTOOL.PUB.ROBELLE

>base STORE,5,READER
>get D-SALES
>if PRODUCT="WIDGET"
>sort CUST-ACCOUNT
>output SALESUB,erase
>xeq
IN=20, OUT=6. CPU-Sec=1. Wall-Sec=1.
Step 3: QUIZ can now read the subfile

- Change the QUIZ report from
  
  >access D-SALES
  >select if PRODUCT="WIDGET"
  >sort on CUST-ACCOUNT
  >Heading ...

  to

  >access *SALESUB
  >sorted on CUST-ACCOUNT
  >Heading ...

- QUIZ TIP: Compiled QUIZ program doesn't require mini-dictionary
Linking multiple data files

Using Suprlink with PowerHouse

- Suprtool reads and selects records from each data file
- Suprtool sorts the qualified records on the link field into flat files
- Suprlink links the files into one record and writes to the subfile
- QUIZ does the final reporting
Linking with QUIZ versus Suprlink

- **QUIZ**
  - Links on key fields
  - One-to-many links
  - Link field appears twice in subfile

- **Suprlink**
  - Links flat files on any sorted field
  - Each input file record can generate only one output record
  - Link field appears once in output record
# Linking with Suprlink versus QUIZ

<table>
<thead>
<tr>
<th>M-CUSTOMER File</th>
<th>D-SALES File</th>
</tr>
</thead>
<tbody>
<tr>
<td>CUST-ACCO</td>
<td>PURCH-DATE</td>
</tr>
<tr>
<td>NAME-FIRST</td>
<td>CUST-ACCOUNT</td>
</tr>
<tr>
<td>NAME-LAST</td>
<td></td>
</tr>
<tr>
<td>10001</td>
<td>19931015</td>
</tr>
<tr>
<td>Darlene</td>
<td>10003</td>
</tr>
<tr>
<td>Hamilton</td>
<td></td>
</tr>
<tr>
<td>10002</td>
<td>19931015</td>
</tr>
<tr>
<td>Gordon</td>
<td>10003</td>
</tr>
<tr>
<td>Lackner</td>
<td></td>
</tr>
<tr>
<td>10003</td>
<td>19931015</td>
</tr>
<tr>
<td>John</td>
<td>10003</td>
</tr>
<tr>
<td>Melander</td>
<td></td>
</tr>
<tr>
<td>10008</td>
<td>19931020</td>
</tr>
<tr>
<td>Thomas</td>
<td>10010</td>
</tr>
<tr>
<td>Serafin</td>
<td></td>
</tr>
<tr>
<td>10009</td>
<td>19931021</td>
</tr>
<tr>
<td>Gordon</td>
<td>10016</td>
</tr>
<tr>
<td>Oxenbury</td>
<td></td>
</tr>
<tr>
<td>10010</td>
<td>19931021</td>
</tr>
<tr>
<td>Wayne</td>
<td>10016</td>
</tr>
<tr>
<td>Humphreys</td>
<td></td>
</tr>
<tr>
<td>10011</td>
<td>19931001</td>
</tr>
<tr>
<td>William</td>
<td>10020</td>
</tr>
<tr>
<td>Kirk</td>
<td></td>
</tr>
<tr>
<td>10012</td>
<td>19931028</td>
</tr>
<tr>
<td>Percy</td>
<td>10020</td>
</tr>
<tr>
<td>Ferguson</td>
<td></td>
</tr>
<tr>
<td>10013</td>
<td></td>
</tr>
<tr>
<td>Colin</td>
<td></td>
</tr>
<tr>
<td>Andersen</td>
<td></td>
</tr>
<tr>
<td>10019</td>
<td></td>
</tr>
<tr>
<td>Rupert</td>
<td></td>
</tr>
<tr>
<td>Hillstrom</td>
<td></td>
</tr>
<tr>
<td>10020</td>
<td></td>
</tr>
<tr>
<td>Walley</td>
<td></td>
</tr>
<tr>
<td>Nisbet</td>
<td></td>
</tr>
</tbody>
</table>

- QUIZ links 6 records; 14 records if optional link

- Suprlink links 3 records; 11 records if optional link
Replacing QUIZ with Suprlink

- Change one-to-many links to many-to-one; output file cannot contain more records than input file
- Field sequence is different from QUIZ output
- Link field is not repeated in output record; record length of Suprlink output file is smaller than QUIZ
- Optional linkage defaults fields to blanks or zeros
Debugging tip

- First create a self-describing (SD) file with the LINK option
  ```
  output SALCUST,LINK
  ```

- Use FORM command to examine record structure
  ```
  form salcust
  ```
  ```
  File: SALCUST.HANS.TECHSUP (SD Version B.00.00)
  Entry: Offset
  CUST-ACCOUNT Z8 1 <<Sort 1>>
  DELIV-DATE I2 9
  PRODUCT-NO Z8 13
  PRODUCT-PRICE I2 21
  PURCH-DATE I2 25
  ...
  POSTAL-CODE X6 135
  Limit: 108 EOF: 8 Entry Length: 140 Blocking: 29
  ```
Creating subfiles from multiple datasets

- Indexed link

  QUIZ

  > Access D-SALES link to M-CUSTOMER
  > report summary CUST-ACCOUNT NAME-FIRST &
  >       NAME-LAST PURCH-DATE
  > set subfile name ...

  QTP

  > Access D-SALES link to M-CUSTOMER
  > subfile SALFILE size 10000 keep &
  >       include D-SALES, NAME-FIRST, NAME-LAST
  > set input limit 0
  > go

- Suprlink does not repeat link field in output record
Creating complex subfiles without an indexed link

- Suprlink can link files on any field
- QUIZ requires an index to link

How do you create a QUIZ subfile with the required fields?

Two steps:
1. Build a one-record subfile of each data file
2. Link subfiles on record number to create new subfile
   - Access *SALSUB link to record(0) of *CUSTSUB
   - Report summary CUST-ACCOUNT NAME-FIRST ...

- Subfiles must have correct item definitions; data not important
Creating subfiles without a PowerHouse dictionary

- Create a one-record subfile with QUIZ

  ```
  >define NAME-FIRST character size 10 = " "
  >define NAME-LAST character size 20 = " "
  >define DELIV-DATE Integer size 4 = 0
  >report summary all
  >set subfile name ... 
  ```

- Ensure data-types match actual data
  - `Integer*4` is not the same as `Integer size 4`
  - Check record length of subfile against data
Creating new data fields

- Suprtool can summarize at sort breaks
  >`duplicate none keys total Sales=total`
- Suprtool creates new fields for totals
  - Field names ST-TOTAL-1, ST-TOTAL-2, etc.
  - Appended to record
  - Field format P28 (packed-decimal)
- To create a compatible field in QUIZ:
  >`define D-Total packed size 14 = 0`
A typical requirement: Summary values

- Create subfile with sort and total fields

- Calculate sort-break totals with Suprtool
  ```
  >get d-sales
  >sort cust-account
  >dup none keys total sales-total
  >extract cust-account
  >out saltot,erase
  ```

- Use totals in QUIZ report
  ```
  >access *saltot link to d-sales
  >define d-pcnt num*3 = 
  (sales-total/st-total-1)*100
  >report .....
Summary of Speeding Up QUIZ

- Suprtool delivers qualified data to QUIZ
- Data must be in a format QUIZ understands
- Use PowerHouse to create its own data structures
- Create new items with QUIZ Define commands
- Use Suprtool FORM command to examine structure
- Use Show Items in QUIZ to display structure

Carefully examine your requirements
Call our toll-free number if you need help
Speed Demon

Working with Speed Demon

- What is Speed Demon? 2
- Speed Demon versus Suprtool 3
- Speed Demon intrinsics 6
- SPDEDBSCAN 7
- SPDEDBINIT 8
- SPDEDBSHUT 11
What is Speed Demon?

- Reads records sequentially
- Same as serial DBGET for user programs
- Useful when you want to extract more than 50% of a dataset
- Works best in 3G languages, such as COBOL, Pascal and FORTRAN
Comparing Speed Demon and Suprtool

- Suprtool is a stand-alone utility program
  - Selects, sorts, and extracts records
  - Puts extracted records in an output file
  - Output is available to application programs for further processing

- Speed Demon is an intrinsic library
  - Extracts records
  - Cannot select or sort records
  - Moves extracted records directly to the application program
Extracting records

Application program -> SPEED DEMON

SPEED DEMON extracts

SPEED DEMON calls

transferred to

Report program

extracts
Versions of Speed Demon

- Compatibility mode (CM) version on MPE V
  - Faster than DBGET
  - Slower than Suprtool
  - Uses a small amount of stack space

- Native mode (NM) version on MPE/iX
  - Faster than DBGET
  - As fast as Suprtool
Speed Demon intrinsics

- SPDEDBINIT selects dataset and field list
- SPDEDBSCAN replaces calls to DBGET mode-2
- SPDEDBSHUT cleans up after dataset scan
- SPDEEXPLAIN prints error messages
SPDEDBSCAN replaces DBGET mode-2

SPDEDBSCAN intrinsic

- Has similar parameters to DBGET, but without NO SET, MODE or LIST options
- Must call SPDEDBINIT before SET and LIST
- Mode-2 serial access is always assumed
SPDEDBINIT selects dataset and field list

SPDEDBINIT intrinsic

- Mode-1 returns the entire record
- Mode-2 allows you to specify a list of field names
- Required before each SPDEDBSCAN
- Scans only one dataset at a time
- Requires database opened with DBOPEN command
SPDEDBINIT in mode-1 returns complete record

- Call "SPDEDBINIT" using:
  - `db-base`
  - `db-set-d-sales`
  - `db-mode-1`
  - `db-status-area`
  - `spde-db-control`
  - `db-dummy-arg`.

```plaintext
01 spde-db-control.
   05 spde-db-version pic s9(4) comp value 0.
   05 spde-db-buffer pic s9(4) comp value zeroes.
   05 spde-filler pic x(20) value spaces.
```
SPDEDBINIT in mode-2 returns specific fields

- Speed Demon accepts all valid TurboIMAGE field lists except "*" list

- Move "CUST-ACCOUNT,PRODUCT-NO, PRODUCT-PRICE;" to db-list-d-sales.

- Call "SPDEDBINIT" using db-base
  
  db-set-d-sales
db-mode-2
db-status-area
spde-db-control
db-list-d-sales.
SPDEDBSHUT cleans up after DBSCAN

SPDEDBSHUT intrinsic

- Mode-1 closes the database
- Mode-2 closes the database and prints a performance report
- If omitted, will cause next SPDEDBINIT to fail

Call "SPDEDBSHUT" using db-base
  db-set-d-sales
db-mode-1
db-status-area
spde-db-control
db-dummy-arg.
Summary

- Replacement for DBGET mode-2
- Intrinsics