LOADING A DATA WAREHOUSE WITH LEGACY DATA

Clive Cooper, Department of Social Welfare, Wellington, New Zealand
Claire Somerville, Team Comtex, Wellington, New Zealand

ABSTRACT

Over a period of three years, three hundred Gbytes of historical data were extracted from an operational database and stored on 3,000 IBM® 3480 tape cartridges. This data is used for management reporting and research purposes. As part of the process of building a data warehouse on a large UNIX server, this data was to be moved to on-line storage on the new platform.

Consolidation of the historical data was necessary to reduce the on-line disk storage space required. The consolidation process used proc compare to remove redundant data which had not changed from one time period to another.

This paper gives an overview of the project to implement a data warehouse, followed by a description of the process of converting and consolidating a high volume collection of historical data.

THE DATA WAREHOUSE

Data warehouse implementation is one of the current hot items in the information technology systems area. The data warehouse environment must provide for the inclusion of data which originates internally or externally, from numerous source systems, and in many formats. There may be large amounts of data involved. The time period covered by the data may also be extended, encompassing data from older ‘heritage databases’ through to the present. SAS® provides powerful tools in these environments because of its ability to read data in so many different formats.

The warehouse provides a repository supporting a complete separation of the business operational data systems from the decision support information environment. It contains all the data needed to manage an enterprise. It also supports the concept of data in one place, avoiding the problems of multiple (and often differing) copies of data distributed throughout a business.

These business issues resulted in the Department of Social Welfare (DSW), New Zealand, establishing a project to implement a data warehouse. This paper gives some insight into one aspect of the implementation: the transfer of a significant volume of historical data from heritage information systems onto the new Information Analysis Platform (IAP), which is the department’s name for its data warehouse.

To put that part of the data warehouse project into context, this paper will cover:

- The department’s business structure
- The operational and information data environments
- The IAP platform and some of the specialized software

- An overview of the transfer of the major history data to the IAP
- A detailed description of the consolidation of the historical data using SAS proc compare.

BUSINESS STRUCTURE

The department is structured into a number of business units under a single permanent head, who has a contract of service provision with the Minister of Social Welfare. Each business unit has a General Manager.

The business units are:

- New Zealand Income Support Service (NZISS)
- Children and Young Persons Service (CYPS)
- Community Funding Agency (CFA)
- Social Policy Agency (SPA)
- Tritec (providing computer services).

A Corporate Services group provides centralized and common services to the business units.

OPERATIONAL COMPUTER SERVICES

The operational computer services within the department are provided on a number of different platforms:

- Unisys®: a triple A19 with a mirrored DMS II database for the SWIFTT (benefit payments) and TRACE (debt management) systems used by NZISS
- IBM: a 3090 under MVS/KA used to operate the department’s financial management systems (VSAM files); and support the use of SAS for the major information analysis work against snapshot subsets (known as the ‘monthly dump’ of the SWIFTT data (of which much more later)
- Several vendor’s UNIX machines; running Oracle® databases for CYPS, CFA and the Corporate Human Resources system.

The NZISS operational system, SWIFTT, operates in a Unisys DMS II environment. It is not practicable to access that database for data analysis. The approach has been to take a monthly snapshot or dump of the main table structures. This data is then converted to SAS data sets on an IBM 3090 and used for further
analysis. It is this historical SWIFTT data which is to be moved to
the IAP.

INFORMATION ANALYSIS PLATFORM

The specification for the business requirements for the platform
was issued, responses evaluated, and the major components
ordered.

The RFP sought a prime vendor to supply a UNIX platform to
support:

- An Oracle database environment to replicate
  SWIFTT and other operational databases
  (estimated to require some 60 Gbytes of
  storage)
- The historical monthly dump data (about 300
  Gbytes in total volume) available on-line
- SAS software was to be the analysis
  environment
- End user "point and click" reporting tools.

IAP IMPLEMENTATION

There was no single vendor providing a completely satisfactory
response to the RFP. A major difficulty was the cost of providing
for the large volume of historical data on-line. The approach
chosen was to source the available components of a solution
from the offerings in the tender responses and have
implementation of the overall IAP project managed by the
department. The implementation would include the consolidation
of the historical data from 300 Gbytes to a forecast 60 Gbytes.

Database replication software for the DMS II Unisys environment
was licensed from Unisys and hardware was purchased from
Hewlett-Packard. The database replication software provides
facilities for the extraction of a complete copy of the original
DMS II database and then the extraction of database changes
from the database audit logs. The software also produces an
SQL script to create a copy of the original database under Oracle.
Using this software it will be possible to maintain the IAP
database up to date within a maximum of 24 hours of the source
database.

The hardware platform is a three processor HP9000/K400, with
one Gbyte of RAM. There are 300 Gbytes of disk storage
provided by 10 disk arrays, each containing 15 two Gbyte disk
units. Three autoloader DAT tape drives are provided for file
system backup and an autoloader 3480 tape cartridge reader for
loading the large amounts of data from the Unisys and IBM
mainframe environments onto the platform. The platform
operating system is HP-UX 10.01. The database environment is
Oracle 7.2.

DATA CONSOLIDATION OBJECTIVES

The first objective of this part of the project was to move the
historical SWIFTT monthly dump data from the IBM mainframe
environment and transfer it to the new UNIX platform. This data
consisted of 45 structures, with up to 37 months of historical data
for each structure.

SWIFTT data had not been archived since the system came live
in 1991. As a result, each month’s dumped data contained a lot
of redundancy; much of one month’s dumped data was a direct
duplication of the previous month’s data.

There was an estimated 300 Gbytes of historical data to be
moved to the new environment. The business units needed this
historical data stored on-line. This quantity of disk storage would
be available on the new platform, but was not available solely for
historical data – the historical data disk storage requirements
would have to be reduced.

This then was the second objective of the project: to reduce the
amount of disk storage required on the new platform by removing
the redundant data.

A listing of the files was obtained from the mainframe, but very
little documentation existed on these historical files. It was
known that problems existed with the data. These problems
resulted from, among other things, the shifting responsibility of
data creation following restructuring within the department: no
one business unit had responsibility for the data integrity of the
dumped data. Some files were known to be unreadable; some
contained invalid data; some files were incomplete; in many
cases the structural content of the files had changed over the
years but these changes were not documented.

The project had to identify and document the known, and as yet
unknown, changes to the data. This documentation would be
necessary to ensure the successful conversion of valid data to
the new platform, and would also be incorporated into the Oracle
CASE data dictionary now being established.

MOVING THE DATA

The first stage of the project was the physical move of the data
from the IBM mainframe environment to UNIX. The historical
data was stored in SAS data set format on an IBM mainframe
running SAS 6.08. It was to be moved to the UNIX environment,
which was running SAS 6.09.

There were three steps in moving the data from the IBM to the
UNIX environment:

- XPORT prepare on IBM/MVS
- Transfer
- XPORT restore on UNIX.

As a first step, the data was converted into transport format on
tape. Proc copy was used with the xport engine:

```
//SERN DD DSNAME=S.SOC.STATS.SBEN.D110392, 
// UNIT=TAPE, DISP=SHR
libname xsben xport
 't.soc.iap.xport.sben.d110392' unit=cart 
 disp=(new,catig)
proc copy in = sben out = xsben ;
run ;
```

At the same time, a proc print of a few records and a proc
contents was run. These procs were used, along with the
number of observations recorded in the proc copy procedure, to gain some documentation and information on the data on each tape. In many cases, there was more than one file for any one month, with no existing documentation as to which was the correct one. Empty data sets, and invalid files were identified during this process of creating transport files, and file content changes were documented.

A total of around 1,500 tapes were created during this process, using over 33 hours of IBM 3090 CPU time. There was some saving in size in the transport phase. For example, a 900 Mbyte file was converted into a 863 Mbyte transport file – a 4% saving.

The tape cartridges containing the transport files were then physically moved to the new platform site. REEL Exchange software was used to read the tape to a disk library on the UNIX, with a Storage Tek 4280 cartridge unit, equivalent to the IBM 3480. Proc copy was again used to copy the files from transport format to SAS data set format on disk, and the data was backed up to DAT tape.

CONSOLIDATING THE DATA

The aim of the consolidation process was to reduce the amount of disk storage required on the new platform, by the removal of all redundant data.

The consolidation process could have been done on the IBM mainframe, but the volume of data involved and the disk restrictions in that environment meant that the consolidation would have to be done on the UNIX, where 300 Gbytes of space would initially be available for this work.

The consolidation process would compare each month’s data with the previous month’s data for each structure, and identify any changes or differences from one month to the next. A file of changed records would be created from each comparison. The first file for any structure would be used as the base data set, and subsequent files of changed records would be added to it. In this way, one structure at a time, all redundant data would be removed.

PROC COMPARE

Proc compare can be used to compare the contents of two SAS data sets: the base data set and the compare data set. A file of differences can be output.

The basic syntax is:

```
proc compare <option-list>
  var variable-list;
  with <descending> variable-1 <...<descending> variable-n.
  <notsorted>;
  by variable-list;
run;
```

To test if the base data set and the compare data sets are the same, proc compare does the following:

- Compares the data set attributes which are set in a data statement
- Compares the variables in one data set with the variables in the other data set
- Compares the attributes of the matching variables, including type, length, and format
- Checks that each observation in one data set matches an observation in the other data set.

Proc compare outputs the results of this comparison in:

- Printed reports
- Messages in the SAS log
- Output data set
- Numeric return code found in macro variable &sysinfo

Options

Proc compare has 41 different statement options. These options can be grouped into five functions:

- Specify the data set names
- Specify the content of the output data sets
- Control comparisons of data values
Database Management Facilities

- Control detail in printed reports
- Control listing of variables and observations

Output Files

Different output files can be obtained from proc compare. Output can be specified to contain any of the following:

- Observations from the base data set
- Observations from the compare data set
- An observation for each pair of matching observations showing the difference between the variable values
- An observation for each pair of matching observations showing the percent differences between the matching variable values
- Or all of the above
- Statistics

Options can be set to output only records in the base data set which differ from the compare data set, or records in the compare data set which differ from the base data set. Or both can be output.

The dif type observation is a record which is created for each pair of matched records in the base and compare data sets. The value for each variable shows the difference between the values in the pair of observations.

The output data set could contain four records for each difference found. For example, the base record, compare record which is different from the base record, a dif record containing a "mask" showing the difference between the two records for each variable, and a percent record showing the percent differences.

Statistics

Summary statistics can be produced for variables being compared which are judged to be unequal. The following statistics are produced for numeric variables:

- Number of nonmissing values
- Mean of values
- Standard deviation
- Maximum value
- Minimum value
- Standard error of the mean
- Probability of greater absolute T value if the true population mean is 0
- Number of matching observation judged unequal, and percent unequal
- Difference between the mean of the base and comparison values
- Correlation of the base and comparison values

- Square of the correlation of base and comparison values

For character variables, the only statistic produced is the number of matching observations which are judged unequal.

Reports

A variety of reports can be produced, or suppressed. The default report includes:

- Data set summary
- Variables summary
- Observation summary
- Values comparison summary
- Value comparison results for variables

Additional report printout can be obtained with the printall option, or suppressed with the noprint option.

PROC COMPARE TESTING

The data was known to contain discrepancies and inconsistencies. Either the files would need to be standardised before being compared, or the output change files would need to be standardised before being appended to the consolidated data set. Standardisation of the data sets would entail ripples back changes through the files, so that the first file would assume the same structure as the last.

Data changes included the following: some fields changed their name or size; new fields were added; old fields were deleted; fields changed from numeric to character; lengths and labels had been added. Consolidation of unstandardised files would have to ensure that comparisons were made of compatible data.

Proc compare was tested to verify that it could be used to satisfy the data consolidation required, and to identify which options would be used.

Some testing was conducted on small data sets on a PC/LAN, other testing was done on large data sets on the IBM mainframe.

The following tests were designed to test some of the known data differences.

New Records and Updated Records

Using proc compare, records in the base data set which are not in the compare data set may remain in the base data set and not be output to the update data set. Records in the compare data set which are not in the base data set can be output to the update data set.

Variable Name Changes

Variables with different names but which contain the same data can be compared by using var old-variable; with new-variable; syntax.
Variables in One Data Set

New variables, for example in the compare data set which are not found in the base data set, are ignored by proc compare. All other fields are compared, and only records with changes in the other fields are output to the update data set. A decision has to be made at this point whether to add or loose the new variables.

Changed Attributes

Proc compare ignores changes in field labels, length or formats and outputs changes in field values. Differing attributes are reported on in the output.

Changed Field Types

Some fields have a change of type — for example from character to numeric. In this case, proc compare issues a warning of variables with inconsistent types in the log and does not compare these variables. All other variables are compared and changes output to the update data set.

Variable Size Change

Some variables have changed their size in the data sets — for example a reason code field changed from $2. to $3. Proc compare will compare these two variables and output observations with changed values.

Mainframe Testing

For this project two files of around two million records each were taken, and proc compare was run over these two files. This test showed certain discrepancies which testing of smaller data sets on the LAN had not revealed.

The importance of the sort sequence was identified. If the two files contain the same multiple records for a single social welfare number for example, but the records are in a different order, then proc compare will not be able to correctly compare these records. An id statement containing the sort order variables was required to resolve this problem.

If one file contains an additional record for a single social welfare number and there are no id variables, then proc compare gets out of step and finds all subsequent records are different.

Processing of large files was complicated by the absence of unique keys.

Summary Reports

The problem with the summary reports is to obtain the right amount of output — it is easy to get too much or too little. If an id variable is used then the summary reports are produced by id variable. If a file contains one million social welfare numbers, and the id variable is the social welfare number, then the result is a voluminous output which we had to suppress.

Retaining some statistics did treble the output, but it also helped identify data discrepancies.

Proc Compare Syntax

Though proc compare could cater for many of the known data changes, it was decided to standardise the files before comparing them.

The following proc compare syntax would be used:

```sas
proc compare
  base = month1-data-set
  compare = month2-data-set
  out = update-data-set
  outnoequal outcomp outbase nomiss stats novalues ;
  var _all_;
  id sort-order-variables ;
run ;
```

The options used in this statement include:

- `nomiss` Judges missing values as equal — for example missing values set to "A" will be found equal to missing values set to ".".
- `stats novalues` Reduces the printed output — care needs to be taken that the printed output does not include a record for every record found in the base and compare data sets
- `outcomp` Outputs observations from the compare data set which are not found in base.
- `outbase` Outputs observations from the base data set not found in compare.
- `outnoequal` Outputs records which are not judged to be equal in the two data sets.

Base records were output and used to match back to the consolidated records, to fix a replacement date on the "deleted" records. The change records were appended to the consolidated file after each compare.

CONSOLIDATED FILES

Each consolidated file had three additional fields added: a file date, replacement date, and a key sequence number. The file date field identified when the record was first found in the monthly dump. The replacement data indicated that a record had either been replaced by an update, or deleted and not found in a subsequent dump. The key sequence was created to provide a unique key.

By using these two dates, users could reconstitute any month’s data from a consolidated file by using the following macros. These macros were added into the SAS macro library.
The extract macro reconstitutes one month's data:

```
%macro extract (reqdtd);
  if filedate le &reqdtd'd and
    (repdate = . or
     repdate gt &reqdtd'd)
  %mend;
%extract (ddmmmyy);
```

The get_then macro gives the user the flexibility to extract more than one month's data and add in additional processing in one pass of the data:

```
%macro extract (reqdtd);
  if filedate le &reqdtd'd and
    (repdate = . or
     repdate gt &reqdtd'd) then
  %mend;
%extract (ddmmmyy);
```

Large files which were greater than the two gigabyte limitation were split by the first key field. Reconstituting the data required setting both consolidated data sets for the structure.

CONCLUSION

The topic covered in this paper has shown the type of activity needed to make the important historical data available. It must be remembered that in its early period of operation it is the history which will receive most use, as the database replication will have little more than today's current data.

In this project we were confronted with large quantities of high volume historical data and the need to move it from SAS data set tape files on an IBM mainframe, to a new Information Analysis Platform based on a UNIX box. The historical files were known to contain a high level of redundancy in the form of duplicate data. The disk storage space available on the UNIX required the consolidation of the historical data.

Proc compare was chosen as a viable method to extract changed data differences, reducing the space requirements on the UNIX to an estimated 60 Gbytes.

The contribution of the SAS software in the establishment of this platform so far has been significant. It is most unlikely that custom built procedures in any other language could have been developed and applied to the problem in the same time frame.

References


Author Contact:

Clive Cooper
Data Administrator
Information Systems Co-ordination Unit
Department of Social Welfare
Private Bag 21
Wellington
New Zealand
Phone 64 4 472 7666
Fax 64 4 472 3608
email clic@actrix.gen.nz

Clare Somerville
Senior Consultant
Software Solutions
Team Comtex
PO Box 2390
Wellington
New Zealand
Phone 64 4 471 5600
Fax: 64 4 472 6796
email clares@actrix.gen.nz

SAS is a registered trademark or trademark of SAS Institute Inc. in the USA and other countries. Unisys is the registered trademark or trademark of Unisys Corporation. ™ indicates USA registration.

Other brand and product names are registered trademarks or trademarks of their respective companies.