Understanding SAS/Warehouse Administrator®

Presented by Michael Davis,
Bassett Consulting Services, Inc.
on Monday, March 31, 2003
at the SUGI 28
Seattle, Washington

Purpose

- make the case for using WA over traditional methods for creating and maintaining data warehouses and marts
- illustrate how data warehouses and marts are modeled in WA, highlighting the creation of metadata
- share and document tips and techniques for using WA

Data Warehousing is…

- process of making operational data available to decision support applications (such as SAS!)
- involves extracting, transforming, joining, sorting, summarizing, and consolidating operational data

Why Data Warehousing ?

- subset raw information information
- merge header and detail records
- sort, index reporting tables
- summarize (tables and OLAP)
- standardize code schemes
- standardize physical file structures
- cleanse "dirty" data

Why Data Warehousing ?

- reduce intra-organization discrepancies
- reproducible reporting results
- document data repository
- document warehousing processes
- improve performance of operational applications
- save human resources

Why WA ?

- "point and click" interface
- changes are automatically posted to generated SAS code
- generates process flowcharts
- generates HTML documentation
- load-sharing and dependent job scheduling via LSF JobScheduler
- availability of process libraries and other features to structure the warehousing process

What is Metadata ?

- information that defines sources, data stores, code libraries, and other resources
- used to write the actual SAS code
- technical metadata defines where the data lives and how to access
- business metadata defines what the data means, who is responsible for it

Why Metadata ?

- single point of control
- documentation across platforms
- eases changes and maintenance because WA generates SAS code
- available to search and report
- can import and export metadata to other applications

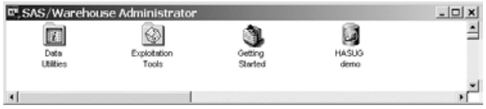
Other Vocabulary

- data *warehouse* verses data *mart*
- ETL Process is what the code generated and launched by WA actually does
- visit Data Warehousing community on SAS web site for more data warehousing vocabulary
- web tutorial is very helpful in understanding how to set up that first warehouse in WA

Starting WA

- run from within SAS session as a SAS desktop application
- start from Solutions pmenu or "DW"
- right-click to open new warehouse
- double-click on existing warehouse to open it

Starting WA (con't)

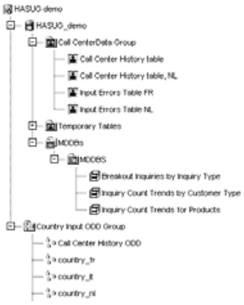


After entering DW on the command line

Environment Hierarchy

- Data Warehouse Environment
  - Data Warehouses
    - Subjects
      - Data Groups
      - Infomarts
      - OLAP Groups
  - Operational Data Definition Groups
    - Operational Data Definitions

Environment Hierarchy (con't)

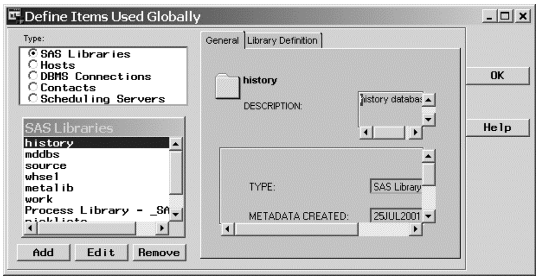


- HASUG demo
  - HASUG_demo
    - Call CenterData Group
      - Call Center History table
      - Call Center History table, NL
      - Input Errors Table FR
      - Input Errors Table NL
    - Temporary Tables
    - MDDBs
      - MDDBS
        - Breakout Inquiries by Inquiry Type
        - Inquiry Count Trends by Customer Type
        - Inquiry Count Trends for Products
  - Country Input ODD Group
    - Call Center History ODD
    - country_fr
    - country_it
    - country_nl

Global Metadata

- SAS libraries
- hosts (local, remote)
- DBMS connections (V8 may be libref)
- contacts
- scheduling servers

Global Metadata (con't)



Define Items Used Globally

Type:
● SAS Libraries
○ Hosts
○ DBMS Connections
○ Contacts
○ Scheduling Servers

SAS Libraries
history
mddbs
source
whse1
metalib
work
Process Library - _SA

history

DESCRIPTION: history databa

TYPE: SAS Library

METADATA CREATED: 25JUL2001

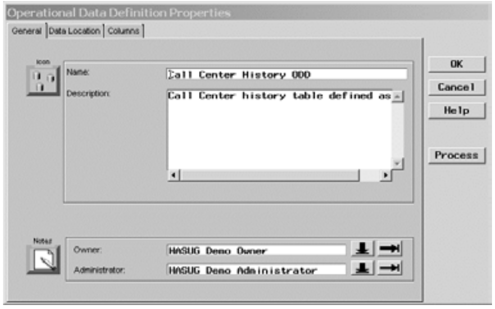
Typical SAS Libraries

- DBMS engine librefs
- detail data
- source
- metalib (_DWMD)
- process library (_SASWA)
- whse1, whse2, etc.
- work

Operational Data Definitions

- General tab includes description and owner
- Data Location tab includes host, library, and table name
- Column tab has "contents" information
- Tip: import columns from existing table
- Tip: after resorting, "Save Order"

Operational Data Definitions (con't)



Operational Data Definition Properties

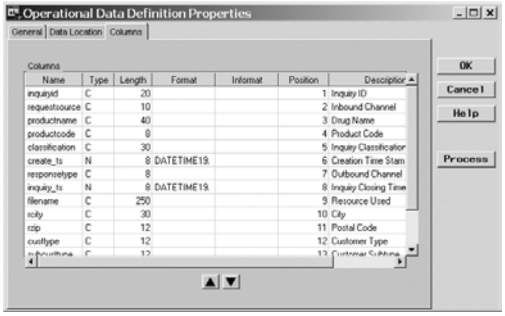
Name: Call Center History ODD

Description: Call Center history table defined as

Owner: HASUG Demo Owner

Administrator: HASUG Demo Administrator

Operational Data Definitions (con't)


Operational Data Definition Properties

Columns

Name	Type	Length	Format	Informat	Position	Description
inquiryid	C	20			1	Inquiry ID
requestsource	C	10			2	Inbound Channel
productname	C	40			3	Drug Name
productcode	C	8			4	Product Code
classification	C	30			5	Inquiry Classificatior
create_ts	N	8	DATETIME19.		6	Creation Time Stam
responsetype	C	8			7	Outbound Channel
inquiry_ts	N	8	DATETIME19.		8	Inquiry Closing Time
filename	C	250			9	Resource Used
rcity	C	30			10	City
rzip	C	12			11	Postal Code
custtype	C	12			12	Customer Type
subcusttype	C	12			13	Customer Subtype

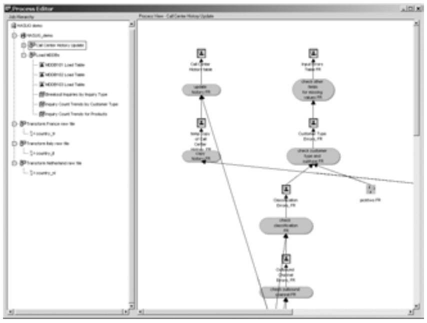
Environment – Next Step ?

- additional ODDs
- target output structures:
  - data tables
  - MDDB cubes
  - files (flat files, CSV, ODS destinations)

Process Editor

- the window is used to create and manage WA jobs
- consists of two panels:
  - Job Hierarchy
    - Job Groups
    - Jobs
  - Process View
    - shows process flow

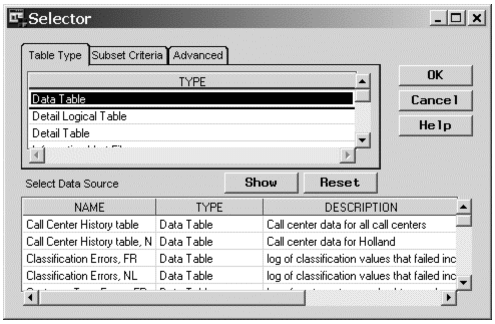
Process Editor (con't)


Process Editor
Call Center History Update
Load MDDBs
MDDB101 Load Table
MDDB102 Load Table
MDDB103 Load Table
Breakout Inquiries by Inquiry Type
Inquiry Count Trends by Customer Type
Inquiry Count Trends for Products
Transform France raw file
Transform Italy raw file
Transform Netherland raw file

Process View Panel

- Tools -> Process Editor
- flow is from bottom to top, left to right
  - plan the flow before inserting objects
- first add the process output(s)
- then define the input tables(s)
- same physical table can be both an output table and an input table

Sample Output Table Selector


Selector

TYPE
Data Table
Detail Logical Table
Detail Table

Select Data Source

NAME	TYPE	DESCRIPTION
Call Center History table	Data Table	Call center data for all call centers
Call Center History table, N	Data Table	Call center data for Holland
Classification Errors, FR	Data Table	log of classification values that failed inc
Classification Errors, NL	Data Table	log of classification values that failed inc

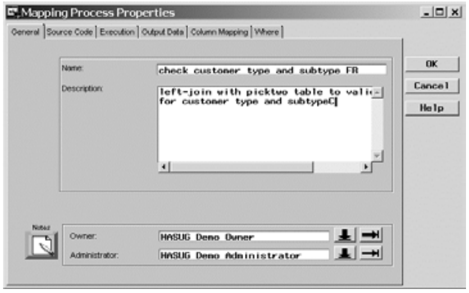
What is a Mapping Step ?

- defines how columns and rows from the input tables are mapped to the output tables or MDDB cubes
- can specify one-to-one, one-to-many, or many-to-one mappings
- generated SAS code is PROC SQL
- add-in tools can customize behavior

Mapping Step Tabs

- General has name, narrative, owners
- Source Code is WA generated, user-written, or supplied by process library
- Execution can be local or other host
- Output Data defines destination
- Columns Mappings are 1:1 or derived
- Where describes row filtering

Mapping Step Tabs (con't)



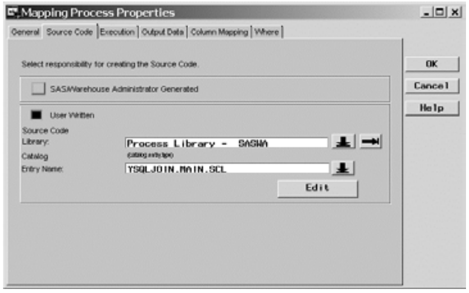
Name: check customer type and subtype FR

Description: left-join with picktwo table to vali
for customer type and subtypeC

Owner: HASUG Demo Owner

Administrator: HASUG Demo Administrator

Mapping Step Tabs (con't)



Select responsibility for creating the Source Code.

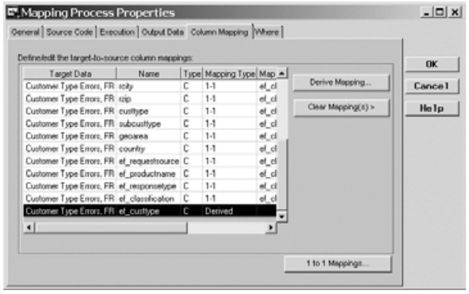
□ SAS/Warehouse Administrator Generated

■ User Written

Source Code Library: Process Library - SASWA

Catalog (catalog.entry.type) Entry Name: YSQLJOIN.MAIN.SCL

Mapping Step Tabs (con't)



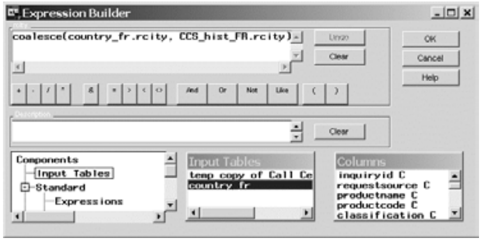
Define/edit the target-to-source column mappings:

Target Data	Name	Type	Mapping Type	Map
Customer Type Errors, FR	rcity	C	1-1	ef_cl
Customer Type Errors, FR	rzip	C	1-1	ef_cl
Customer Type Errors, FR	custtype	C	1-1	ef_cl
Customer Type Errors, FR	subcusttype	C	1-1	ef_cl
Customer Type Errors, FR	geoarea	C	1-1	ef_cl
Customer Type Errors, FR	country	C	1-1	ef_cl
Customer Type Errors, FR	ef_requestsource	C	1-1	ef_cl
Customer Type Errors, FR	ef_productname	C	1-1	ef_cl
Customer Type Errors, FR	ef_responsetype	C	1-1	ef_cl
Customer Type Errors, FR	ef_classification	C	1-1	ef_cl
Customer Type Errors, FR	ef_custtype	C	Derived	

Case Expressions

- use to define conditional mappings
- can "pick and click" via the Expression Builder, avoids typing errors
- syntax is CASE … WHEN … THEN … ELSE … END
- left-join using Coalesce function
  - coalesce(update table, history table)

Case Expressions (con't)



```
coalesce(country_fr.rcity, CCS_hist_FR.rcity)
```

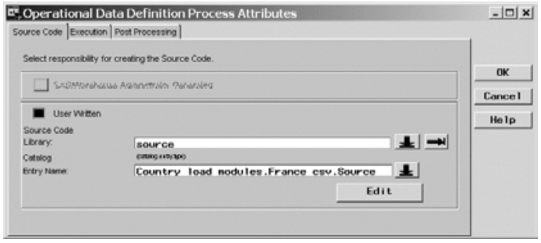
Input Tables:
- temp copy of Call Ce
- country fr

Columns:
- inquiryid C
- requestsource C
- productname C
- productcode C
- classification C

Load Steps

- are where to put user-written code to override WA-generated SAS load code
- problem is that user-written code is a "black box", not reflected in metadata
- goal is to model the process in WA, letting WA generate the SAS code
- tip: do not forget to specify host!

Load Steps (con't)



Select responsibility for creating the Source Code.

□ SAS/Warehouse Administrator Generated

■ User Written

Source Code

Library: source

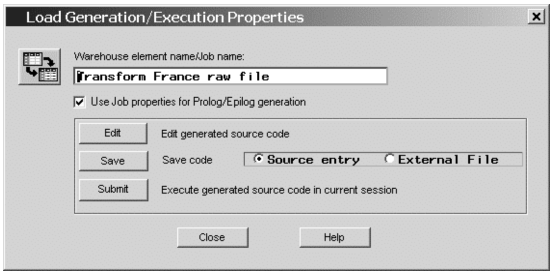
Catalog [illegible]

Entry Name: Country load modules.France csv.Source

Executing the Job

- right-click -> Run
- right-click -> View Code -> All to generate code for external scheduler
- to schedule jobs through WA, need to set up a scheduling server and job information library
- may need SAS SAS/SHARE for job information library

Executing the Job (con't)



Warehouse element name/Job name: Transform France raw file

☑ Use Job properties for Prolog/Epilog generation

Edit — Edit generated source code

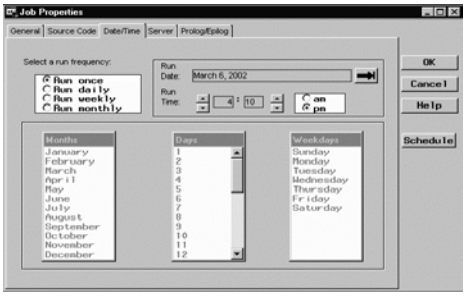
Save — Save code ● Source entry ○ External File

Submit — Execute generated source code in current session

Scheduling Jobs in WA

- WA can send jobs natively to CRON, AT
- WA can send jobs via the null scheduler to LSF JobScheduler via LSF add-in
- use LSF for dependent jobs and load-sharing
- LSF is OEM licensed to the platform on which WA is installed, requires keys

Scheduling Jobs in WA (con’t)



Select a run frequency:

● Run once
○ Run daily
○ Run weekly
○ Run monthly

Run Date: March 6, 2002

Run Time: 4 : 10 ○ am ● pm

Months	Days	Weekdays
January	1	Sunday
February	2	Monday
March	3	Tuesday
April	4	Wednesday
May	5	Thursday
June	6	Friday
July	7	Saturday
August	8	
September	9	
October	10	
November	11	
December	12	

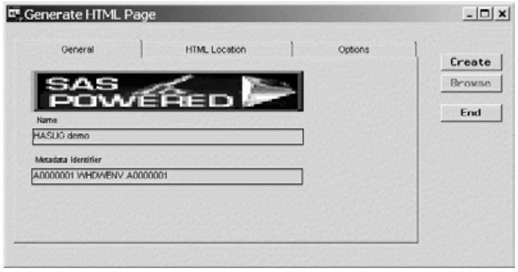
Add-In Tools

- installed on top of WA in Release 2.2
- provide additional tools to help load external data, model processes, schedule jobs, and to analyze, search, and report on metadata
- write custom add-in tools using SCL
- changes with Data Builder

Generating HTML Documentation

- done via an add-in tool
- lists:
  - columns
  - input sources
  - dependencies
  - source code
- allows web sharing of documentation

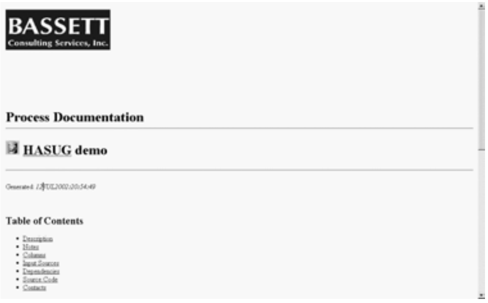
HTML Documentation (con’t)



Name: HASUG demo

Metadata Identifier: A0000001.WHDWENV.A0000001

HTML Documentation (con’t)



Process Documentation

HASUG demo

Generated: *13JUL2002:20:54:49*

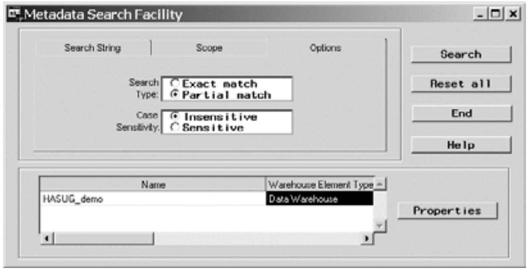
Table of Contents

- Description
- Notes
- Columns
- Input Sources
- Dependencies
- Source Code
- Contacts

Metadata Search Facility

- Tools -> Search Metadata …
- enter search string
- limit search by elements and fields
- does not have to be an exact match

Metadata Search Facility (con’t)



Search Type: ○ Exact match ● Partial match

Case Sensitivity: ● Insensitive ○ Sensitive

Name	Warehouse Element Type
HASUG_demo	Data Warehouse

Migrating Metadata

- problem: path embedded in metadata
- copy using Metadata Copy wizard
- tip: map metadata to same drive letter and path as network copy

WA Changes & Enhancements

- multiple-table join
- takes advantage of multiple-threading
- File Import Wizard is integrated
- “open metadata”

Data Builder

- replaces Warehouse Administrator
- open metadata available to SAS, other apps
- understands RDBMS and ERP metadata
- check in/check out eliminates SAS/SHARE
- server architecture enables remote operation
- keep open multiple Java windows and jobs
- DB can convert existing WA jobs

Friendly Advice for April 2003

- Data Builder requires SAS Management Console and Open Metadata Server
- SMC and OMS are Release 9.1 features
- consider Release 9.1 availability and anticipated reliability (e.g. add-in tools)
- possible strategy is to use WA now, run DB in parallel, and convert when ready

Conclusion

- explained advantages of WA over traditional methods for creating and maintaining data warehouses and marts
- illustrated how data warehouses and marts are modeled in WA and highlighted the creation of metadata
- shared some tips and techniques for using WA

Contact Information

Michael Davis
Bassett Consulting Services, Inc.
10 Pleasant Drive
North Haven CT 06473-3712
e-mail: michael@bassettconsulting.com
http://www.bassettconsulting.com
telephone: 203-562-0640