

Using SAS® Enterprise ETL Server to Build a Data Warehouse: Focus on Student Enrollment Data

Evangeline Collado, University of Central Florida, Orlando, FL

M. Paige Borden, University of Central Florida, Orlando, FL

ABSTRACT

There is a continued demand across the higher education sector for increased extraction and analysis of university data to facilitate strategic decision-making and management of university programs and activities. Using the SAS® Enterprise ETL Server, the University of Central Florida's Office of Institutional Research has merged more than 10 years and 5.5 million rows of student enrollment data from both legacy file structures and ERP-system file structures, as part of our data warehouse reporting solution. This paper will present the design concepts, ETL coding, data cleansing, metadata and final table structure of the student enrollment data tables.

Once complete the full data warehouse will provide data integrity and integration for a breadth of information applications; improve reporting efficiency and consistency for a community of independent users; while allowing a shift from simple reporting to strategic performance measurement, evaluation, forecasting and decision-making.

WHO WE ARE

MISSION

The mission of the Office of Institutional Research is to provide information that is timely, easily accessible and of the highest quality to enhance decision-making, strategic planning, and assessment at the University of Central Florida.

PURPOSE

Institutional Research (IR) provides electronic and web-based dissemination of official information to the University community (including the Board of Trustees, the various colleges, departments and other academic and administrative units), external agencies, and the Florida Board of Education (BOE); generates, supervises or develops all official University data reports and state-required BOE reports; provides end-user data solutions and training; business intelligence analysis for decision-making purposes. The director and staff serve on numerous university-wide committees and workgroups and assist with the collection and interpretation of institutional data, assist in planning academic programs, and participate in the implementation of evaluative procedures. The functions of the office support the entire university enterprise.

BACKGROUND

Institutional Research has, as one of its primary responsibilities, the task of reporting all official data to internal and external constituents. Staff from the IR office routinely meet with end users and participate in workgroups and committees that deal with data needs and information access. This allows us the opportunity to interact directly with end users and determine, one-on-one, exactly what their data needs are and the format that would best meet their needs. This also has the added benefit of allowing IR staff to more immediately respond to needs and make changes to our reporting environment as they are requested.

INTRODUCTION

The University of Central Florida (UCF), a major metropolitan university with more than 42,000 students, has relied on the Office of Institutional Research to maintain a crude set of data marts that allow for the creation of reports and provide responses to institutional data requests. The dynamic nature of the university and the never-ending demand of performance measurement, evaluation, forecasting and strategic decision-making data needs called for a more complex data warehouse that ensured data integrity and integration. Institutional Research (IR) chose the SAS® Enterprise ETL Server and SAS® Enterprise BI Server to fulfill these university data needs.

IR merged more than 10 years and 5.5 million rows of student enrollment data from both legacy file structures and ERP-system file structures, as part of our data warehouse reporting solution. This paper will present the design concepts, ETL coding, data cleansing, metadata and final table structure of the student enrollment data tables.

STUDENT ENROLLMENT DATA

Institutional Research prepares five student data course files (SDCF) annually for submission to the Florida Board of Education. Three of the files are "final" data and two files are "preliminary" data. The data file contains student biographical/demographical details, degree information, term and career-based records data, course enrollment, fellowship, stipend and waiver information. In the past ten years, these files have been submitted using both legacy source data and ERP-system source data. Although both formats were available electronically, the two source data formats resulted in different data file output structures making trend reporting especially difficult.

DESIGN CONCEPTS

In UCF's legacy system, the student enrollment data was stored in two tables; SDCF_A and SDCF_B. SDCF_B included all the bio/demo information, degree details and term and career-based student record data, one row per student per term. The SDCF_A table included all course enrollment, fellowship and stipend awards, and waiver information, allowing for multiple rows per student, per term.

In UCF's ERP system, the student enrollment data was stored in five tables; SDCF_DATA, SDCF_CRSE, SDCF_FELO, SDCF_STIP and SDCF_WVRS. Although the structure increased from two to five tables, the SDCF_DATA table still contained one row per student per term, and the other four tables allowed multiple rows per student, per term.

We decided to use the design structure of the ERP system for the warehouse – five tables – to allow for a smoother ETL process in the future since additional data will come from the ERP system each term. We felt it was better to keep the fellowship, stipend and waiver data separate from the course data for ease of reporting. Thus, the most difficult ETL process would involve loading the historical data from the legacy system since all of this information was stored in the SDCF_A tables. The target repository would be SAS tables stored on a new server that was purchased specifically for the data warehouse project.

Aside from different table names and structures, the legacy v. ERP data also included format changes in several fields. For example, the student major field in legacy was a 6-digit numerical value. In the ERP data, the student major field is a 10-character text value and the major code field is an 8-character field. Some fields used as primary keys in the ERP system did not have a matching field in the legacy system. For example, in the legacy system we used a student's social security number as an ID variable, whereas, in the ERP system we use an 11 character ID field. We needed to obtain this ID from the ERP system for students that attended UCF beginning from 1995, so we had to join additional tables to the ETL process.

ETL CODING

For ease of use and availability, the legacy data were maintained in Microsoft Access databases. A separate table for each academic year had been created due to software table size limitations. Thus, we had 18 different tables in one database which all had to be extracted, transformed, loaded and appended. The most recent academic year, 2004-2005, would be extracted from the ERP system tables which are stored in an Oracle database.

It was determined that the ETL process for the student data course files would be broken up into two parts: 1) Create intermediate target tables which contain the data from the legacy system; 2) Create permanent target tables which contain all data from the ERP-system and the intermediate tables combined. Then every term each new file will be appended to the permanent target table. This involved creating two ETL jobs to load the historical data and a third job to append each new term in the future. We will keep preliminary data separate from final data and store that in the warehouse following the same 5-table structure.

The first step was to determine which legacy fields mapped one-to-one with which ERP fields, which legacy fields would need some transformation to map to an ERP field and which ERP fields had no matching field in the legacy system but could be mapped in from an alternate table. Once we had an idea of how we would construct the resulting target tables we were ready to begin using SAS® ETL Studio.

It is extremely helpful if you plan your repository structure in advance. Decide what you will name your groups, libraries and tables so they will be consistent, self-explanatory and easy to track. Always group your newly created objects so they do not end up in the UNGROUPED folder. You may end up with more than one object with the same name and delete the wrong item when you try to clean up the repository. Provide clear, concise descriptions for each object to add to the metadata. You should always design your jobs using a bottom-up approach; i.e., build from the target table backward. When defining the columns in the target tables you should import the metadata from a source table whenever possible. For example, if all the columns you want to have in the final target table are defined in several of the source tables used to load the target table then you could choose Import and select the fields from each of the tables and add them to the column list. This saves a lot of time and lessens the possibility of typographical errors. You can make changes to descriptions or other properties while creating the tables also.

We created a library group and a table group for the source data and for the target data. Within the target library group we created one group for the intermediate target library and another for the final dimension target library. The same convention was followed for the target tables group. Source libraries had to be created for MS Access databases and Oracle databases. We planned to use SAS/ACCESS[®] to ODBC software to extract the data from the MS Access tables but had problems with decimal type fields. SAS[®] technical support determined that it was a Microsoft problem so we decided to use the IMPORT procedure in a SAS[®] program to create SAS[®] data sets first and then use that data as the source for the intermediate tables. SAS/ACCESS[®] to PC File Formats software is needed to import data from MS Access or Excel.

We created a job that would append all 9 years of the SDCF_A tables and another job that would append the SDCF_B tables. The intermediate target table created from SDCF_A tables, SDCF_A_1995_2004, would then be used to create four of the final dimension target tables by filtering the data on the SEGMENT_IDENTIFIER field, and the intermediate table created from SDCF_B tables, SDCF_B_1995_2004, would be used to create the final SDCF dimension table, SDCF_DATA_DIM. Figure 1 shows the job to populate the SDCF_A_1995_2004 table. We joined additional source tables to obtain the ERP-system student ID number (EMPLID_SSN) and the term ID (STRM_TERM).

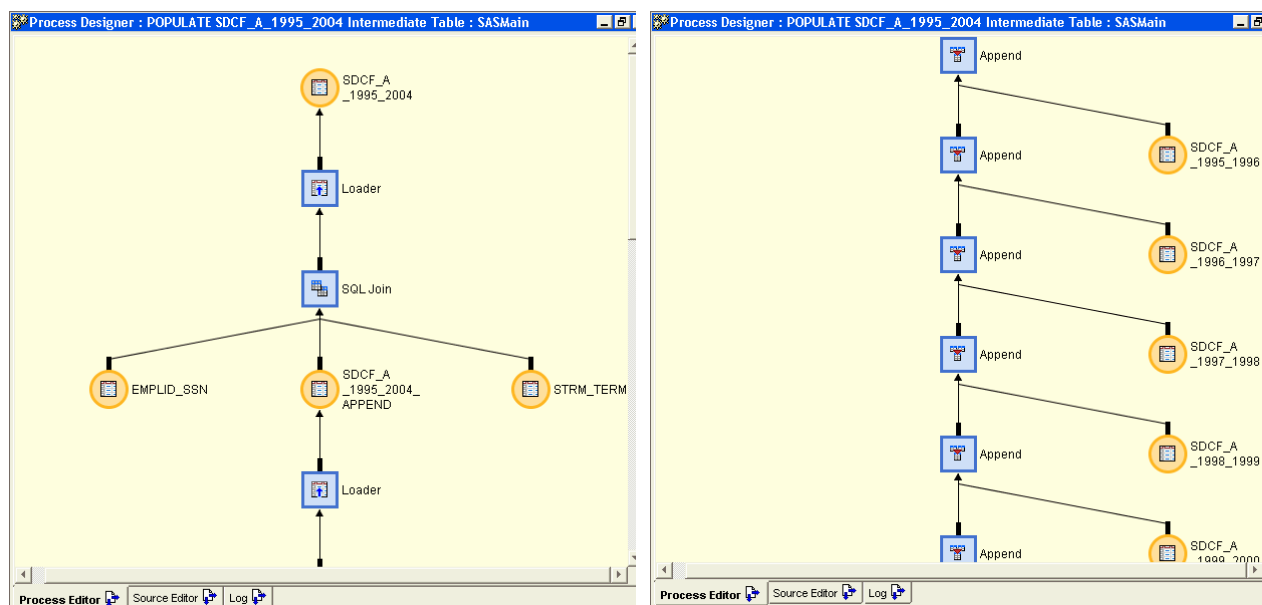


Figure 1

The second job would use Oracle tables so we created a new library that used the SAS/ACCESS[®] to Oracle software. The Oracle client had to be installed on the server prior to the creation of the metadata and we had to obtain the schema name from UCF's Oracle Database Administrator (DBA). When the library was created we had to define a new database server which required that the user and password information be entered in order to connect to the Oracle database. We decided to create a separate SAS Administrator user that had read rights to specific Oracle tables so that none of the staff's user names and/or passwords would have to be used.

Figure 2 shows the diagram for the SDCF_CRSE_DIM table and Figure 3 shows the properties of the SAS Extract where we have filtered the data coming from SDCF_A_1995_2004.

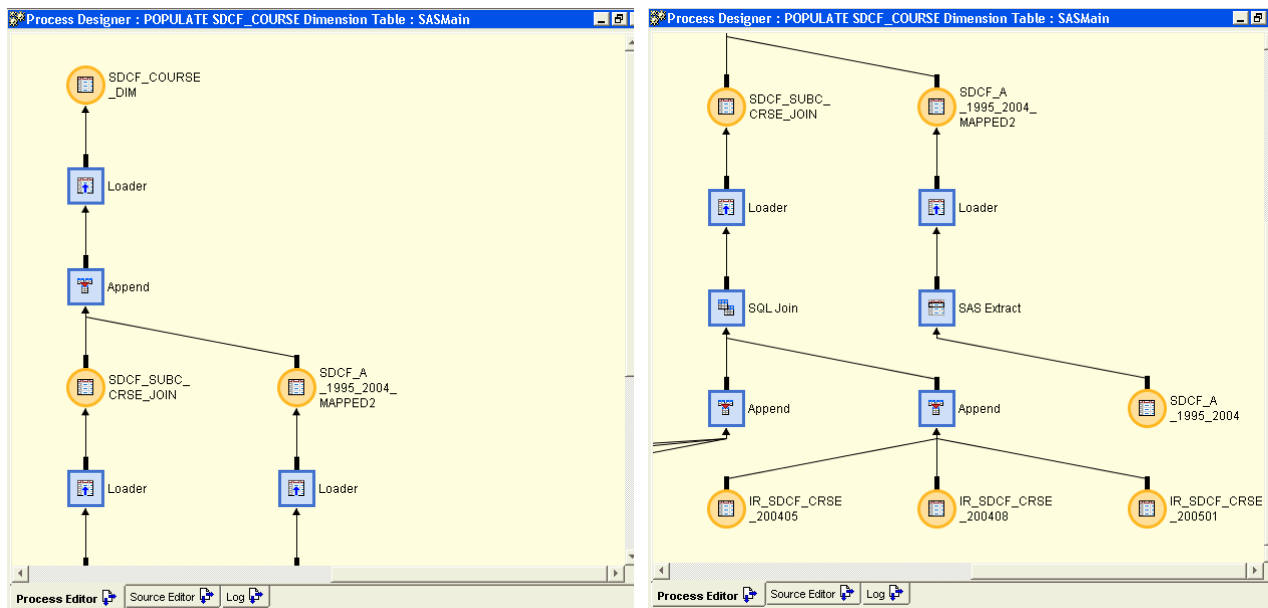


Figure 2

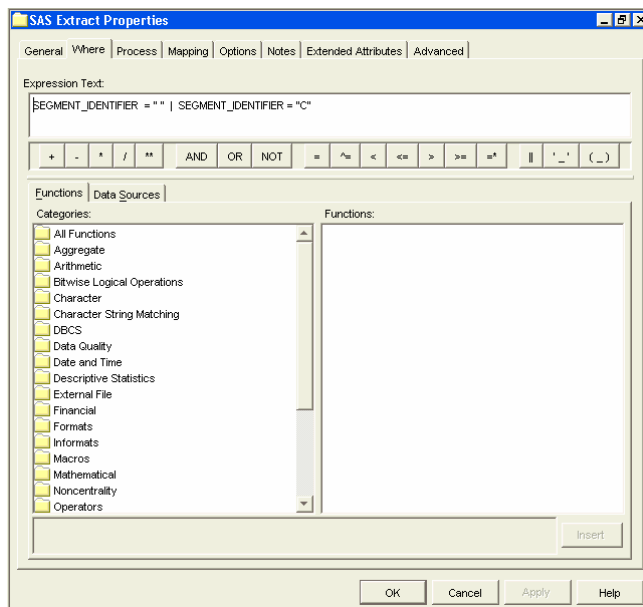


Figure 3

This ETL process was duplicated, with new filter criteria, to create the following final dimension target tables: SDCF_FLWSHP_DIM, SDCF_STPND_DIM and SDCF_WVRS_DIM. The final target table created from SDCF data, SDCF_DATA_DIM was created in a similar fashion using different source tables.

The third job would also use Oracle tables from the ERP system and will append the new term's data to the final target tables created in the previous step. This job will be run once per semester for "final" data and will take place approximately six weeks after the end of the term. The entire process was duplicated for the "preliminary" data and that job will be run approximately six weeks after the beginning of the term.

DATA CLEANSING

One of the biggest challenges we faced was incomplete or inconsistent data values. Some values could be transformed using SAS code, whereas, other values would remain blank if we did not have the information available for that student or term. For example, the SDCF_B tables all have a column named BIRTH_YEAR. However, for the academic years 1995-1996, 1996-1997, and 2000-2001 this field is blank. For the academic years 1997-1998, 1998-1999, and 1999-2000 this field is 4 digits. For the academic year 2003-2004 this field is 2 digits. The ERP system has a BIRTHDATE field which is 10 characters. The optimal solution in this case is to disregard the BIRTH_YEAR field in the final dimension table and map in the BIRTHDATE from the ERP system. Having a complete birth date instead of

just a 2-digit or 4-digit year allows more flexibility and preciseness when trying to calculate a student's age. Besides, you could always get the year only from the entire date with a simple function.

In the legacy system the student's major was a 6 digit field but in the ERP-system the major code was 8 characters. Let's take a look at Computer Science which had the value of "000701" in the legacy system and has a code value of "07.01" in the ERP system. Since some major codes begin with "UC" for Undergraduate Certificate or "GC" for Graduate Certificate before the four-digit code we needed to apply some if-then logic. The following data step was used in the program that imported the data from the MS Access database to create the field with the same values as in the ERP system.

```
DATA libref.tablename;  
  SET libref.tablename;  
  IF (SUBSTR(UCF_MAJOR,1,1)="U" OR SUBSTR(UCF_MAJOR,1,1)="G") THEN  
    ERP_MAJOR=SUBSTR(UCF_MAJOR,1,2)||"."||SUBSTR(UCF_MAJOR,3,2)||"."  
    ||SUBSTR(UCF_MAJOR,5,2);  
  ELSE ERP_MAJOR=SUBSTR(UCF_MAJOR,3,2)||"."||SUBSTR(UCF_MAJOR,5,2);  
RUN;
```

However, in order to get the proper ACAD_PLAN, which is what the major field is in the ERP system, we need to identify the student's academic career since Computer Science is offered for undergraduate, ACAD_PLAN="COMPSCI-BS", as well as graduate study, ACAD_PLAN="COMPSCI-MS" or ACAD_PLAN="COMPSCIPHD" depending on level of graduate. In this case we needed to use several fields to get one value. This can be a tedious process if you have many variables that need to be cleansed or transformed.

Another major data quality issue is data entry. Here at UCF, students apply to the university via a web-based system. They enter information, such as address, city, state, zip code, and phone number onto a web form, which can lead to poor data quality. There can be 50 ways to spell "Orlando" in our database. The SAS® Data Quality Solution will help us integrate and standardize our data across the entire enterprise. We will be able to apply business rules and processes to the metadata such that the resulting data in the warehouse tables will be of the highest quality. The tools provided in this solution are easy enough for non-technical users to cleanse the data which helps keep training costs down. At the time of this writing we have not implemented the data quality portion of our data warehouse project. Information is available, listing all of the key benefits, at <http://www.sas.com/technologies/dw/etl/dqcleansheet/factsheet.pdf>.

METADATA

Although data libraries, data servers, and database schemas can be (and were) defined using SAS® ETL Studio, a component of Base SAS® Software, SAS® Management Console, puts all metadata management tasks together in one easy to use graphical interface. Here is where users and user groups were defined and access rights or authorizations were set. We are in a Windows Server environment so we had to set up the users in Windows first. A word of caution: the "sasadm" user, who is the unrestricted user and created during the software installation process, does not have the authority to view passwords. We struggled with this when we were setting up the Oracle group. After we set up the library to point to the Oracle tables we weren't able to see any data and kept getting an error message. We had to set up the Oracle library while logged on as a restricted user who was specifically given rights to the Oracle data source. We are still learning about the security aspects but feel confident that we have a safe and secure environment.

The SAS® Management Console provides the SAS® administrator with the ability to manage and create repositories from the desktop eliminating the need to be physically located where the server is. All administrative tasks directly related to the warehouse repository are easily managed with this tool. However, when creating the ETL processes that will build the tables in the warehouse, it is much easier to define the libraries within SAS® ETL Studio. The graphical interface is very easy to understand and the wizards take you through the process step-by-step. If you have a carefully constructed plan you should be able to create the ETL processes in a very short time. Of course, there is a trial and error period, so allow extra time in the beginning to think things out.

FINAL TABLE STRUCTURE

After all the jobs were created for the student data course file data we ended up with five dimension tables that now contained more information than we originally had. By incorporating the ERP system data elements we can now provide more appropriate reports without having to create formats or mapping tables for the coded values. The original legacy system data had only codes for gender, ethnicity, student type, etc. but the new warehouse tables also contain descriptions for these codes. This makes it much easier for users to create custom reports that meet their exact needs. They don't need to contact the IR office to ask them what the meaning of the ethnic code "O" stands for. This will not only give the university community faster access to the data but also allow the IR office staff to concentrate on more in-depth analyses and support applications.

CONCLUSION

The one big lesson learned is that it "ain't as easy as it sounds." We thought we had an indestructible plan of execution when we started this project. However, we had to spend some time "cleaning up" the repository when we

were finished. We learned that all Microsoft applications do not work with all SAS® software, but because SAS is what it is or who they are, we were able to find a fast solution and keep moving forward with the project.

We learned that it is extremely important to have the software installed and configured correctly and to apply all hot fixes, patches, and service packs to keep SAS® software working efficiently. We also learned that once metadata is defined for a specific job, if you change anything about the tables in the job it will have a large impact on other processes. It is important to run an Impact Analysis to see just where that one table is used. Finally, we learned how exciting it is to finally have all the data brought together in one place where we can manage, manipulate, analyze and report it to the entire university.

Once complete the full data warehouse will provide data integrity and integration for a breadth of information applications; improve reporting efficiency and consistency for a community of independent users; and allow a shift from simple reporting to strategic performance measurement, evaluation, forecasting and decision-making. The data will be of the highest quality as it is cleansed and transformed based upon consistent definitions. By using the same source to generate all university reports, the level of confidence in the results is extremely high. The warehouse provides less complex reporting and data access since both trained and untrained users will be able to generate the same results on an ever-increasing list of data points.

REFERENCES

The following SAS publications were invaluable tools used extensively during the creation, implementation and maintenance of the data warehouse.

Cameron, Freda, et al. *The Value of Integrated Metadata: SAS Open Metadata Architecture*. (A SAS White Paper) Cary, NC: SAS Institute, Inc.

SAS Institute Inc. (2004), *Introduction to the SAS® Business Intelligence Client Tools, Course Notes*. Cary, NC: SAS Institute, Inc.

SAS Institute Inc. (2004), *SAS/ACCESS® 9.1.3 for Relational Databases: Reference*. Cary, NC: SAS Institute, Inc.

SAS Institute Inc. (2004), *SAS/ACCESS® 9.1 Supplement for Oracle (SAS/ACCESS for Relational Databases)*. Cary, NC: SAS Institute, Inc.

SAS Institute Inc. (2004), *SAS® 9.1.3 ETL Studio, User's Guide*. Cary, NC: SAS Institute, Inc.

SAS Institute Inc. (2004), *Using SAS® ETL Studio to Integrate Your Data, Course Notes*. Cary, NC: SAS Institute, Inc.

ACKNOWLEDGMENTS

We would like to thank Larry Yang, Robert L. Anderson, Keri-Ann Brown, Alfred Tadros and Brett Perlman from SAS for their hard work and assistance getting this project started. We would also like to thank the technical support team at SAS for the assistance provided when things were not working as expected.

CONTACT INFORMATION

Your comments and questions are valued and encouraged. Contact the authors at:

M. Paige Borden
Email: mborden@mail.ucf.edu

Evangeline Collado
Email: ecollado@mail.ucf.edu

Office of Institutional Research
University of Central Florida
P.O. Box 160021
Orlando, FL 32816-0021
Work Phone: (407) 823-5061
Fax: (407) 823-4769
Web: <http://www.iroffice.ucf.edu>

SAS and all other SAS Institute Inc. product or service names are registered trademarks or trademarks of SAS Institute Inc. in the USA and other countries. ® indicates USA registration. Other brand and product names are trademarks of their respective companies.