SESUG 2022 Paper 146

Using SAS® Data Integration Studio as an Effective Data Virtualization Tool

Raj Bhosale ME, MBA, PMP, NC State University

ABSTRACT

Data Virtualization is a process of merging data from diverse sources/systems to create one or more meaningful data sets that reside in a common location. These data sets aid towards creating dashboards, which can serve as a critical tool for making data-informed decisions. There are several challenges in this process including, but not limited to, data type differences, frequency of data refresh, handling unstructured data, de-normalization and performance overhead. While developing a project that built interactive dashboards in an academia setting, I was able to overcome several of these challenges using SAS® Data Integration Studio. Transformations such as Extract, Join, Append, and LASR Analytic Loader along with SAS® expressions like PUT, INPUT, SUBSTR and CASE helped me in creating clean data sets, which fed into a SAS® Visual Analytics Dashboard. This paper gears towards all levels of SAS users (Beginners to Advanced) and will demonstrate how SAS® Data Integration Studio benefits as a powerful Data Virtualization tool and how it can drive SAS® Visual Analytics Dashboards to provide useful information to the end user.

INTRODUCTION

The primary step in creation of any kind of report or dashboard often involves gathering information from various sources. This information then further needs to be analyzed and converted into a standard that is based upon specific business requirements and scope of work. Reports and dashboards can have a variety of audiences and their view of looking at the data can change based upon their roles in the organization. For example, the Board of Directors and other members of the Leadership Team might be interested in aggregated numbers whereas someone at a managerial level might need to see row level data. You can use Data Virtualization effectively in this regard where you create a dataset by merging data from various sources, convert it to the required norms and bring it into a central location. This data set(s) can then serve as a shared source for various reports and dashboard built across the organization.

This paper will demonstrate Data Virtualization technique used in building the 10-Year Student Enrollment/Application/Degrees Dashboards that provide aggregated numbers and trends further broken down by academic structure (College/Program/Degree) and demographics (Gender/Ethnicity). This project uses SAS® Data Integration Studio to combine and massage the data in order to create Virtualization data sets. These data sets flow into a centralized location, which then serve as source data to the 10-Year Dashboards.

This paper will be broken down into three major steps:

1. Data Collection
2. Data Manipulation
3. Data Upload

DATA COLLECTION

This step mainly focuses on looking at the business requirement and determining the different sources of data. As an example, here is the overview of the business requirement that we had for the Enrollment Dashboards:

1. Create a Dashboard to display distinct Enrollment headcounts for the past 10 years.
2. Ability to drill down by various elements like College, Program, Degree Type, Full-Time/Part-Time etc.
3. Ability to drill down by demographic elements like Gender, Ethnicity, International Status etc.

The next step in this process was to identify different data sources/tables and how to bind this data together, in order to create a single data set that feeds as a primary data source towards the dashboard. The Student Information System serves as a primary data source for this step. Under this data source, the following tables are of particular interest:

1. Census Enrollment Table: This table holds term-by-term information for students captured at the Census date of Enrollment (Census date is the last day where a student can make changes to enrollment). Primary elements used from this table are credit hours registered, academic load, Program, Degree Type etc.
2. Application Table: This table contains Application information for all students with flags such as Applied, Admitted, and Enrolled etc. to identify whether an applicant is admitted and enrolled into their program.
3. Degrees Table: This table holds information about what Degrees were awarded to students. This table has a field called completion term that tells when a student was awarded with a degree.
4. Bio Table: This table contains term-by-term demographic information for students. Gender, Ethnicity, Tuition Residency, Country of Citizenship are some of the elements coming from this table.
5. Term Table: This table contains term begin and end dates for each term.

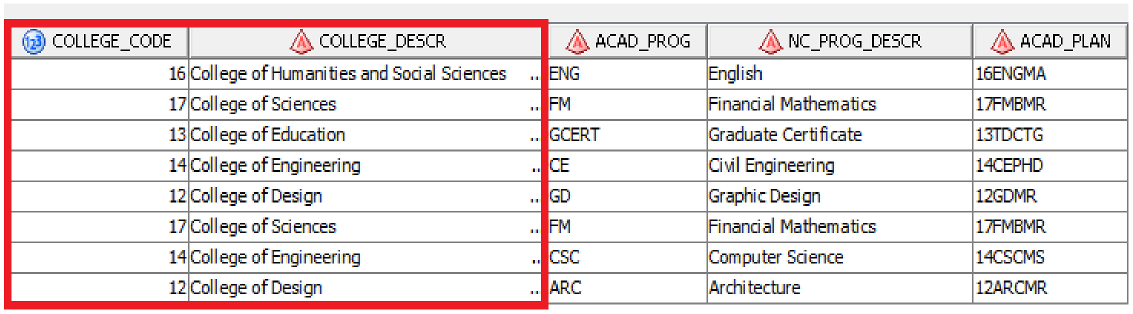
DATA MANIPULATION

This was the most difficult and challenging step out of the three. There were several challenges that we identified during this process. Let us address a few of them and also look at the respective solutions:

1. **Identifying interdisciplinary programs:** Although some programs might have a similar name, they do not necessarily affiliate with the same college. For example, the Biomedical Engineering Program can be a part of College of Engineering and College of Sciences as well. We identified a field called as Academic Plan, which is an alphanumeric value. For example, 14BMEMR. 14 stands for College code, BME stands for Program and MR stands for Degree. We used a case statement here to separate the College, Program and Degree from the Academic Plan. Here is an example of the case statement to separate College:

```
Case
When substr(ACAD_PLAN ,1,2) = '15' Then 'College of Natural Resources'
When substr(ACAD_PLAN,1,2) = '16' Then 'College of Humanities and
Social Sciences'
When substr(ACAD_PLAN,1,2) = '32' Then 'Registration and Records'
When substr(ACAD_PLAN,1,2) = '12' Then 'College of Design'
When substr(ACAD_PLAN,1,2) = '11' Then 'College of Agriculture and Life
Sciences'
When substr(ACAD_PLAN,1,2) = '17' Then 'College of Sciences'
When substr(ACAD_PLAN,1,2) = '18' Then 'Wilson College of Textiles'
When substr(ACAD_PLAN,1,2) = '13' Then 'College of Education'
When substr(ACAD_PLAN,1,2) = '24' Then 'University College'
When substr(ACAD_PLAN,1,2) = '02' Then 'Provosts Office'
When substr(ACAD_PLAN,1,2) = '20' Then 'Poole College of Management'
When substr(ACAD_PLAN,1,2) = '19' Then 'College of Veterinary Medicine'
When substr(ACAD_PLAN,1,2) = '14' Then 'College of Engineering'
ELSE 'ERROR'
End
```

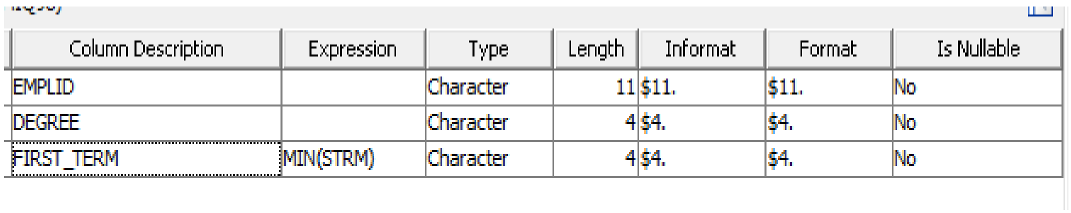
Through this, we were able to separate interdisciplinary programs from each other. For e.g., 14BMEMR belongs to College of Engineering whereas 17BMEMR belongs to College of Sciences. The SAS® function SUBSTR identifies first two characters from the Academic Plan and defines the College accordingly. Here is a screen capture from output of SAS® Data Integration Studio's Extract Transformation:



COLLEGE_CODE	COLLEGE_DESCR	ACAD_PROG	NC_PROG_DESCR	ACAD_PLAN
16	College of Humanities and Social Sciences ..	ENG	English	16ENGMA
17	College of Sciences ..	FM	Financial Mathematics	17FMBMR
13	College of Education ..	GCERT	Graduate Certificate	13TDCTG
14	College of Engineering ..	CE	Civil Engineering	14CEPHD
12	College of Design ..	GD	Graphic Design	12GDMR
17	College of Sciences ..	FM	Financial Mathematics	17FMBMR
14	College of Engineering ..	CSC	Computer Science	14CSCMS
12	College of Design ..	ARC	Architecture	12ARCMR

Display 1. Screenshot of the output of Extract Transformation where College Code and Description are derived from Academic Plan

2. **Calculating accurate "Time to Degree":** For accurate time that a student took to finish their degree, we needed to look at begin and end dates for a student's degree. We looked at the Census Enrollment Table and used SAS® function MIN to calculate what their minimum term of enrollment was for a certain degree. The field named STRM refers to the term. Therefore, MIN (STRM) grouped by degree calculates their starting term. Here is a screenshot of the Extract transformation:



Column Description	Expression	Type	Length	Informat	Format	Is Nullable
EMPLID		Character	11	$11.	$11.	No
DEGREE		Character	4	$4.	$4.	No
FIRST_TERM	MIN(STRM)	Character	4	$4.	$4.	No

Display 2. Screenshot of the mapping on Extract Transformation where MIN (STRM) is derived and grouped by Student ID and Degree

Now we looked at the Degree and the respective Completion Term field from the Degrees table. Next step was to join both these extracts with the Term table to get the Degree Begin Date and Degree End date.

Here is an example of how the Degree Begin and End Dates are obtained using Left Outer Joins:

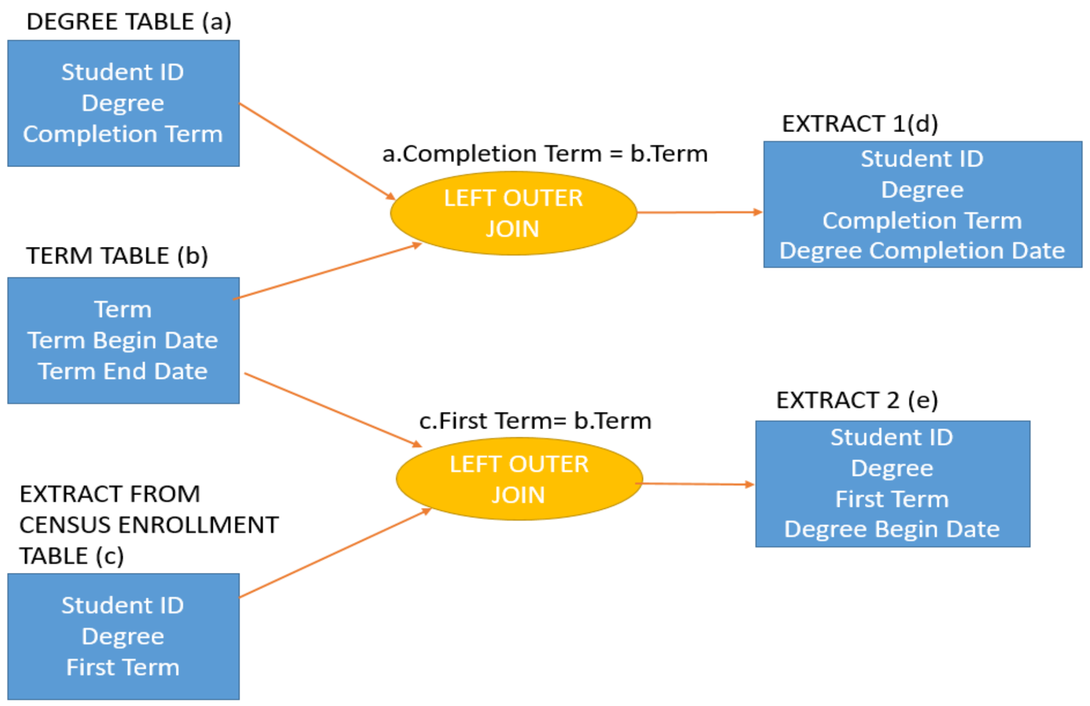


Figure 1: Display of how Left Outer Joins are used to obtain Degree Begin and End Dates

Extract 1 and 2 from the above figure are now further used to calculate the Time to Degree.

Here is a demonstration of how Time of Degree is achieved:

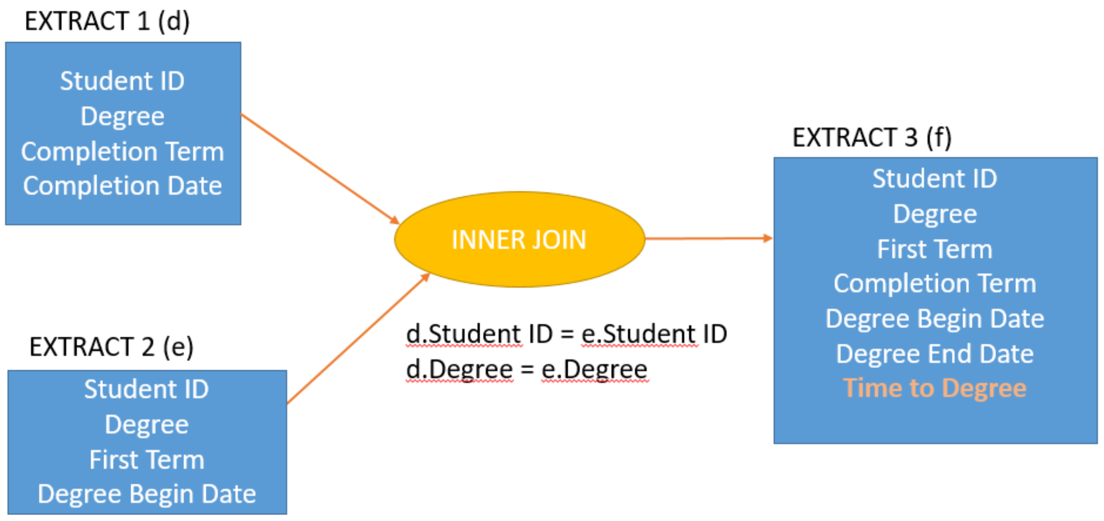


Figure 2: Display of how Time to Degree is calculated using Begin and End Dates

As seen above, the extracts are further joined using an Inner Join. The resulting extract now gives a table that defines a student's Degree Begin and End dates. The field Time to Degree in orange is a calculated field. The following formula was used to calculate the Time to Degree:

```
ROUND ((INTCK ('DTDAY', Degree Begin Date, Degree End Date)/364), .1)
```

The above formula calculates the date difference between the end and begin dates in years rounded to one decimal point.

3. **Data conversion between character and number:** One of the dataset involved combining tables from different systems. We needed Research and Teaching Assistantship data from the HR System which then needed to be merged with the student's academic data from Student Information System. The Student Information System has a field called "Academic Year" which has a data type of character. Similarly, the HR system has a field "Fiscal Year" which is of numeric data type. Also, Fiscal Year runs one year ahead of the Academic Year. The business requirement was to output data in terms of Academic Year so we have to convert Fiscal Year into Academic Year and also change the data type from numeric to character.

   The following formula was used to convert Fiscal Year into Academic Year:

   ```
   PUT((Fiscal Year – 1),4.)
   ```

   Here, the Fiscal Year field was first shifted to (Fiscal Year – 1) as it runs a year ahead of Academic Year. Once that was achieved, the numeric data type was converted to a character data type in order to match it to the data type of Academic Year. Using this technique, the respective extracts from HR System and Student Information System were able to merge with each other.

   The following diagram shows how HR and SIS tables are combined together using left outer join:

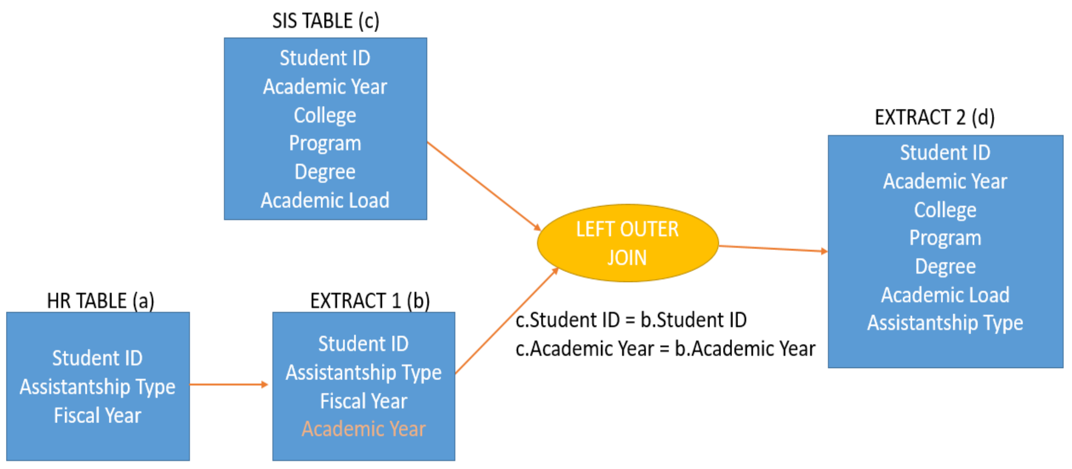
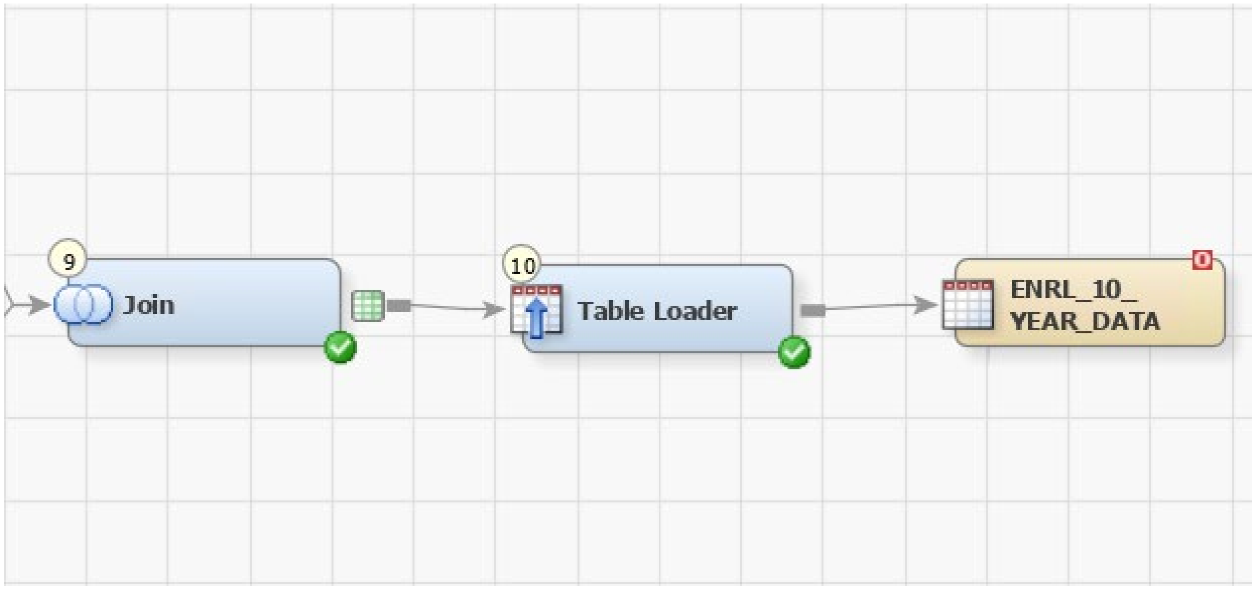


Figure 3: Display of how Fiscal Year is converted to Academic Year and then used in a join to merge data from two different systems

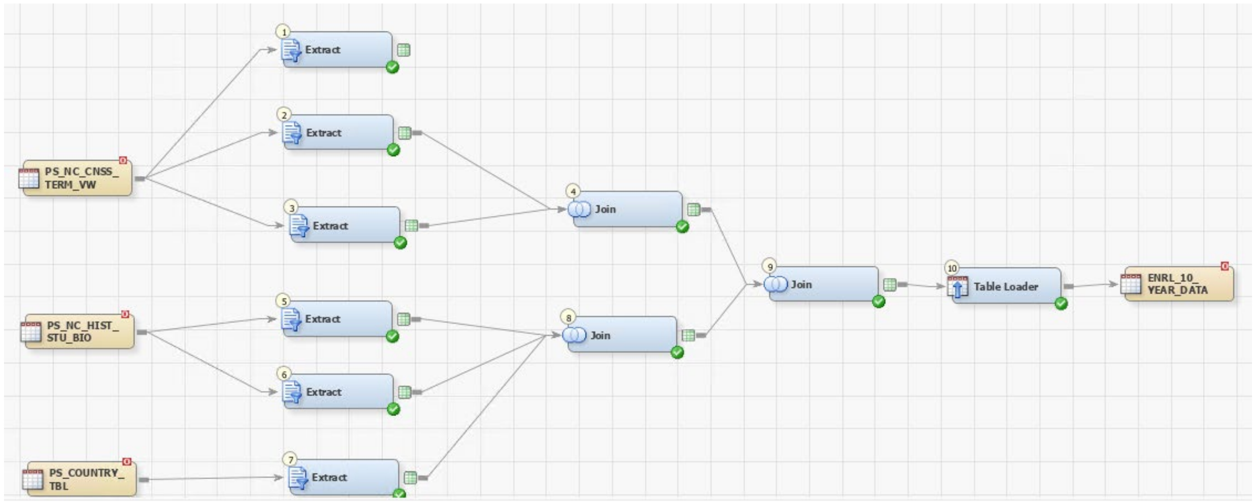
A point to note here is since this is left outer join, the result set is a list of all the students actively enrolled for a particular academic year. The "Assistantship Type" field only gets populated if a student has an assistantship during a given academic year.

After tackling several such challenges, we were finally able to build a stable dataset that was ready to be fed into the SAS® Visual Analytics Dashboard environment. Before we moved on to the Data Upload step, this final dataset was cross verified for data consistency and integrity. Once everything looked good, the working table was pushed into a registered Oracle table using the "Table Loader" transformation. Here is a screenshot of the last step of the Data Manipulation job under the SAS® Data Integration Studio:



Display 3. Table Loader Transformation that inserts data into the Registered Oracle Table

And finally, here is a screenshot of the entire Data Manipulation job from start to finish:



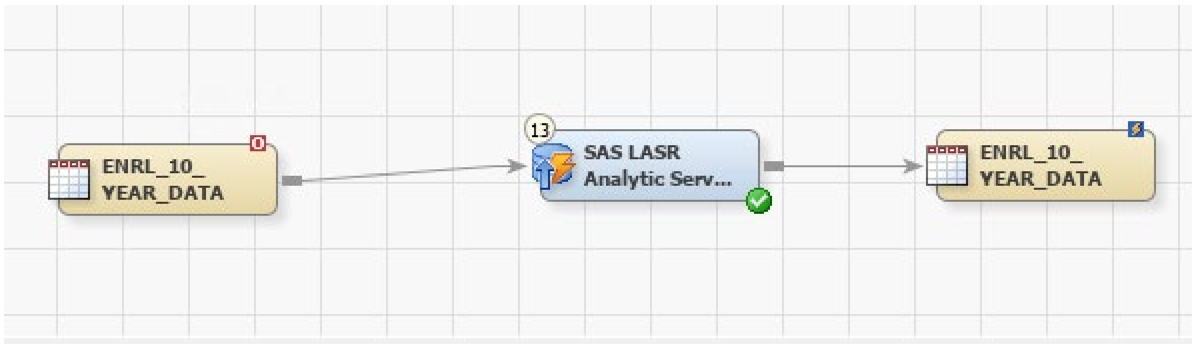
Display 4. Data Manipulation Job for Enrollment

Please note that this is just the Enrollment job. Similar jobs were created for Admissions and Degrees. The respective datasets were fed into the 10-year Admissions and 10-year Degrees Dashboards.

DATA UPLOAD

The final step was to upload this data to the SAS® LASR Server. The registered table that was created in the previous step was then pushed into the SAS® LASR Server environment using the LASR upload transformation under the SAS® Data Integration Studio.

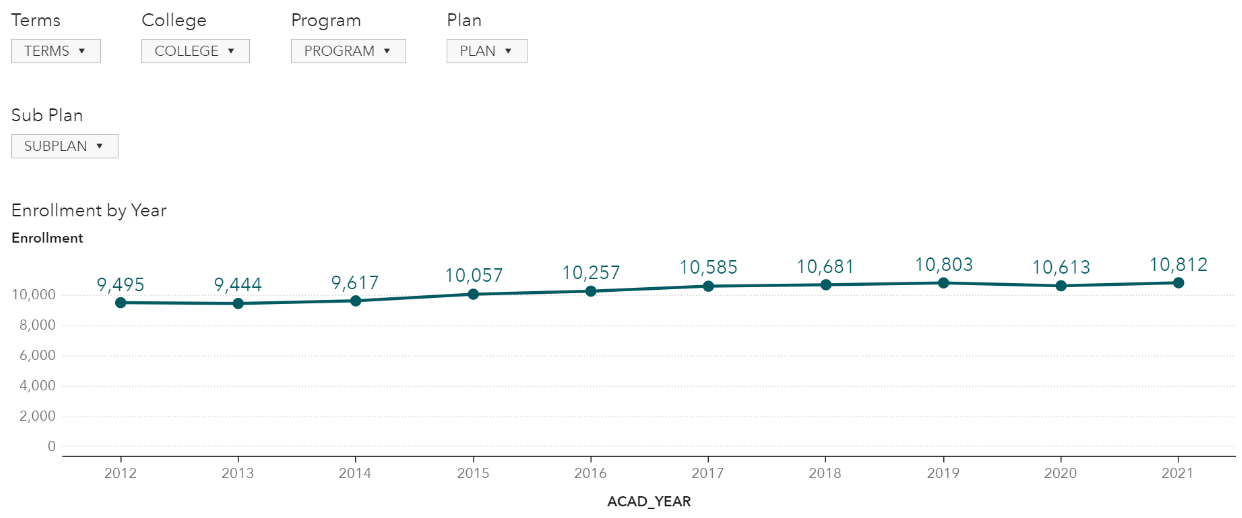
Here is a screenshot of how this step was achieved using SAS® Data Integration Studio:



Display 5. Screenshot of how the data was uploaded to SAS® LASR

This job was then put under Scheduler that runs on a "once per term" basis. The job runs one day after Census Day of Enrollment, which is considered to be the last day for students to drop or add classes; because at that point, all students are locked into their Enrollment schedule and we get to see the numbers for the new term under the dashboard.

Finally, here is a screenshot of one of the pages of the 10-Year Enrollment, which shows a 10 year Enrollment trend:



Display 6. Screenshot of Trends Tab of Enrollment Dashboard

CONCLUSION

As demonstrated above, SAS® Data Integration Studio serves as a very effective tool towards Data Virtualization. With tables from different systems available under a single roof, the data can be successfully massaged and manipulated to the desired output using several transformations like Extract, Join, Table Uploader, Append and many others.

SAS® Data Integration Studio provides the user with a stable dataset that can then be used for several processes like Dashboards, Statistical Models and Ad-hoc reports. After the Graduate School implemented the 10-year Dashboards, we also undertook another project where SAS® Data Integration Studio was used to create a dataset using the "Slowly Changing Dimension" technique. This dataset was used to send automatic emails to different Colleges across the University that contained attached PDF files. These PDF files were able to tell the Colleges how their Admissions and Enrollment numbers have been doing for this year compared to the exact "point in time" for last year.

To conclude, SAS® Data Integration Studio has a huge potential to collect, manipulate and upload data under a central repository which then can be used for several data and analytics processes of the organization.

CONTACT INFORMATION

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

Raj Bhosale (Director of Information Management)
NC State University
pnbhosal@ncsu.edu