**Student Retention Toolkit Using R Dashboard**

**December 3, 2019**

Note: This report contains information which reveals information about Saint Martin’s University database structure and for security reasons should not be released to outside individuals or organizations.

Mary Donahoo

Computer Science Department  
  
mdonahoo@stmartin.edu

Kim Van Vleet  
Computer Science Department  


kvanvleet@stmartin.edu

# Abstract

One of the most frequently requested data by administrators, faculty, and staff are student retention rates and the factors that affect these rates. The rates can be an important indicator of student success and are provided to external agencies and publishers such as U.S. News & World Report for inclusion into their college rankings. Many factors can influence student retention and institutions spend a great deal of resources on how to capitalize or mitigate the effects of those factors. The work done to improve student retention comes from many sides including student success and retention committees, regional and national conferences, as well as companies helping institutions improve student retention.

Many of the factors that impact student retention have been studied in depth but new factors are being discovered. There are also many methods of performing the statistical analysis using student data and providing the information to decision makers in the form of visualizations. These methods depend of the resources available and skill of the staff that retrieve the data and transform the data into information and the effectiveness of the decision makers to apply the information towards improving student retention. In this project or toolkit we retrieved student level data from the Student Management System database using standardized SQL queries and created a repeatable and documented process to import data into the program R. Within R, the Shiny web application was used to create interactive web apps to provide the visualizations.

**Introduction**

Student retention at colleges and universities is an important indicator of student success as is one of the most widely studied areas in higher education [1]. The percentage of students retained one year after they entered college is represented by a retention rate and is used in various college rankings such as U.S. News and World Report. Many factors can influence student retention and institutions spend a great deal of resources determining how to capitalize on or mitigate the effects of those factors. The work done to improve student retention comes from many sides including student success and retention committees, regional and national conferences, as well as organizations helping institutions improve student retention. Many institutions, especially small schools, have limited financial resources and improving student retention can result in a lower cost when students return than it does recruiting new students.

Various methods are used to extract the data that provide the knowledge to help decision makers perform the analysis of student retention. Some larger institutions may have staff members dedicated to retrieving the data with other staff members focusing on the analysis while smaller institutions may have one person retrieving the data and performing the analysis using basic data and software programs. Kroc and Hanson [2] provide a process of understanding student retention. Institutions may initially answer simple questions with descriptive data, then move to more multivariate or qualitative approaches and finally placing the data into a meaningful context for managing student retention.

This paper describes a method for retrieving student data, importing the data into the program R, using the Shiny package for the retention dashboard to provide graphical representations and statistical data needed to support student retention efforts at a small university. There are several other software packages and methods that can provide similar results but we chose R based on its easy to use interface, zero cost, and is taught in several computer science classes on campus.

**Literature Review**

There have been many articles published, seminars, conferences, and specialized companies dedicated to improving student retention. McLaughin and Howard [3] defined three efforts to explain why students leave college: (1) descriptive, (2) comparative, and (3) theoretical. This goal of this project is to assist college administrators with the comparative aspect of why students leave and to help focus efforts on improving student retention.

There were two software systems reviewed by the authors. The first system, GlyphEd (www.glyphed.com), provides software for as little as $99 per month which highlights why students leave using data elements like academics, financials, and engagement through a few, simple visualizations. Naturally, institutions would need to provide a database where those elements are stored and have personnel on-site to provide the expertise to connect those databases. At the higher end, Student SuccessGlyphKIT (Knowledge Insight Kit) contains a collection of visualizations that address students’ retention. It includes two retention views: geo-spatial and non-geo-spatial. The cost range per month is from $450 for the small tier and $1,350 for the large tier. The second system, Tableau (Tableau.com), is very popular among college and university institutional researchers and their booths at the various conferences usually have the most visits by conference attendees. As one example, Tableau demonstrates a “Third-semester Retention Dashboard” that uses data from spreadsheets and runs on Python. The cost for Tableau for a single user is $840 per year, per user, and can either be set up using Tableau-hosted with Tableau Online of self-hosted with a Tableau Server. For organizations needing multi-user access, the Viewer license costs $144 per year, per user, with a minimum 100 users. This package allows for users to view and interact with content on Tableau Server and provides notifications when the data is updated. The Explorer license can connect to any published data source and costs $420 per year, per user, with a minimum of five users. This allows users to author and share content. The Creator license allows the users to connect to any type of data and costs $840 per user, per year. It requires a Creator role provides server administration, manages data flows, and performs the data preparation not unlike the Extract, Transform, and Load (ETL) steps in data warehousing. Some key features of the Creator license are dash-boarding, trend analysis, and logistic regression.

**Design Methodology and Design**

As users can operate with a variety of technology skills sets, environments, and demands, the authors used, as a guide, steps from McLaughlin and Howard’s [3] work in creating management information. The first step, standardized and integrated historical data were defined and restructured to reflect changes necessary for analysis and comparisons. The second step, visualization software required a dataset that was easy to upload while maintaining limited access. Finally, using the toolkit required comprehensive documentation from defining the data in terms of changes in higher education processes that affect the data as well as instructions to the users that own a variety of skills.

In designing the student retention toolkit, the first area of focus was using Microsoft SQL Server Management Studio to retrieve data from the existing Student Information System (SIS) relational database. This database contains snapshots taken at various times of the year for external reporting, the query retrieves data from the most recent snapshot, either from the live, production database, or the nightly backup depending on the access of the user. While many staff have access to various modules, or workflows, in the SIS to enter and view data, only a few staff have access to the underlying database. Utilizing Microsoft’s SQL Server Reporting Services (SSRS) was discussed but since the current Institutional Researcher has read access to the database and can run the query to produce the dataset, the authors decided that adding the SSRS component would involve tasking staff in the Integrated Technology Office. At the present time, it would be a small task for those who have access to the database to simply run the query and have it available for all others users. A future addition would be to leverage SSRS by creating a report that can be formatted into different output types (.CSV, .XLS, .HTML) that can be accessed by more users with different analysis and visualization software packages.

During the design phase, a question was raised about how many years in the past to retrieve the data and what variables to include. Most analysis in student retention is done within the past five years or less. In the current SIS database, the earliest data that can be used consistently is 2004. This was determined by comparing the percentages of values in each variable especially NULL and unknown variables. The SIS client interface was installed and the associated database was created in 2003 and some data was imported from the previous system but not enough for accurate analysis. The query using SQL to retrieve data from the database needed to be robust enough to create consistent variables due to changing data going back to 2004 to the current year. Examples of changing data include revisions to the federal government’s race and ethnicity categories is the fall of 2010, changes to the SAT scores in March of 2016, and changes to practices within the university.

Many attributes are stored for each student and those used in this project are found in various tables depending on what data is stored. For example, the Academics table is usually the basis for most of the student academic attributes. Relationships were created within the query to retrieve data in other tables and views. Some tables are created for each year and term for changing data such as Charge Credit and Residency. Other data objects used are called views because they are select statements which are saved into the database. A view is different than a table because a table contains data. Table 1 below shows the tables and views included in the SQL query.

Table 1. Database tables used to retrieve dataset.  


The second area of focus was determining a no cost and customizable method to provide useful and informative data visualizations available continuously to decision makers. Based on these requirements, the program R and the Shiny web application could provide users with the ability to adjust and select different settings to analyze student retention as well as give the R point-of-contact instructions that they could use to create and customize new data visualizations. A second purpose of the instructions would be to provide continuity to ensure the process remains consistent.

The first steps of the project were to create a planning document that broke the project into several main task areas (Table 2). Using R requires some setup of related components and the detailed instructions are covered later in this report.

Table 2. Main project task lists.



The layout of the web dashboard consists of a main splash page with the Saint Martin’s University logo and a menu on the left which lists the various tools in the retention toolkit (Appendix 1). It includes the visualization charts, data dictionary, glossary, access to the raw data, and a help document. Appendix 2 contains data dictionary which is also included as a menu item on the toolkit dashboard. While the glossary in Appendix 3 describes the variables and how they have changed over time more so for the user, the data dictionary is more useful to those with more of a focus on the database and programming side.

Security was a concern to prevent the identification of students. First, the project is in an academic class setting but utilizes a university student database. One of the preventions of unauthorized entry is to refrain from describing the database architecture such as database or table names. Although access is allowed to only those associated with the database, there is always the threat from hacking and keeping database structure hidden helps to prevent unauthorized entry. Second, providing a link to the raw data in the dataset may allow someone the ability to identify individual students. The more student attributes are provided, the easier it is. Each student has a unique student identification number which allows for easy identification. In the query, a new number was created to act as the dataset’s primary key and prevents the identification of all students. An algorithm was written into the query that would create a 16-bit string that identifies each record but it resulted in duplicate records for some students because of multiple academic records within a single year and term.

**Implementation**

The student cohort data for the project comes from the university’s student information system, which stores data in a SQL Server database. Data is pulled from the database via a complex SQL query. The data is then saved to an Excel spreadsheet, which is subsequently imported into the R application. It’s important to retain the column heading names exactly as they are specified in the SQL query. These heading names are referenced throughout the R code.

To provide an initial dataset to import into R, a small sample of variables were including in the first SQL query. The earliest year that data could be obtained was 2004 which resulted in the data in each variable being verified. A variable, or student attribute, like gender (male or female) was fairly stable longitudinally with either no or a very small number of values other than male and female. As noted above, the categories for race and ethnicities were updated by the United States federal government [4] in the Fall of 2010 and because of a new leading Hispanic or Latino question, which wasn’t asked before, the students’ old race and ethnicity selections couldn’t be translated to the new categories. The old and new categories are stored in different tables in the database resulting in two variables.

Another student attribute that was revised was the SAT [5]. Prior to March of 2016, the SATs consisted of three main sections—Critical Reading, Critical Writing, and Mathematics each with a range of scores between 200 and 800 for a possible total of 2,400. The Critical Reading and Critical Writing SATs were merged producing a new SAT of Evidence-Based Reading and Writing (EBRW). The mathematics component remained unchanged. Both the EBRW and the mathematics have scoring range between 200 and 800 for a possible total score of 1,600.

As variables were added to the SQL query, it became apparent that unknown values were added differently for different variables. For example, only domestic students are asked whether they are first generation or the first in their family to attend college. In the table that stores the first generation response, the field only accepts “Y” for Yes. The value for students who do not select that they are first generation is “NULL”. However, if a “Y” was entered and then removed, the contents changes from “Y” to an empty string which is different than NULL. At the same time, the international office does not ask students if they are first generation. In the SQL query, what would seem to be a retrieval of a simple stored first generation value requires the student’s citizenship. Below is the basic method for retrieving a first generation value from its database table:

1. If not a U.S. Citizen then NULL.
2. If first generation then “Y”
3. Otherwise NULL including changing the empty strings to NULL.

Appendix 4 contains the instructions on accessing SQL Server to retrieve the data set and Appendix contains the SQL code used in the query,

The interactive web application was coded in R using the Shiny dashboard package, which is built on the underlying Shiny package. The basic structure of the Shiny dashboard user interface consists of header, sidebar and main page areas. The sidebar area holds the menus and submenus. Each of the lowest-level menu/submenu items corresponds to its own main page area. A main page area consists of an underlying grid structure which is used to specify the size and placement of objects on the page. These objects can include Shiny input controls such as like radio buttons, check boxes, and slider bars; and outputs such as data visualization plots, data tables, interactive maps, images and text.

The application includes an interactive map that was built using R’s Leaflet package. The map displays the retention rate for the State at Admit. Maps require a shape file of specific geographic information – cartographic boundaries. For example, a shape file might contain the geographic information for all the counties in a state, or for all countries. Because geographic boundaries change over time, shape files are typically available for multiple years. Shape files are available for free from a number of sources. The interactive map requires geographic data of U.S. states, including Alaska, Hawaii, and U.S. territories. The shape file used in the application was downloaded from the following location: https://www.census.gov/geographies/mapping-files/time-series/geo/carto-boundary-file.html

For each of the other data elements used in visualizations within the application, a data frame was created which contained data that has been summarized to provide a count and a percent by the element, year, and retained flag. This summarized data was then filtered and displayed in various ways depending on user input.

**Experimental Result and Analysis**

Much of the testing in the data occurred within SQL but other issues were raised. For example, in the graphical representation, a list is formed containing all of the locations where students originated and the associated retention. However, when incrementing by a year, the list changed if a student didn’t originate from a location where a student originated from the year before.

During testing, that a problem was caused when numerical values in a variable were assigned as “n/a” when the value for some records to be unknown. This resulted in a mismatch of numerical and string data types within a single variable. For these similar variables, NULL was assigned to unknown values which do not provide the data type conflict.

Most of the testing was performed incrementally with the addition of variables in the SQL query to importing the new data into the R environment to producing additional visualization examples.

The ReadOGR() function, part of the ‘rgdal’ package, was used to read the shape file into the R application. Because the project was created for a small university, there isn’t cohort data for every state – even if by using data for all years. For this reason, one of the data preparation steps for the map data is to match the geographic data to the cohort data, eliminating the unmatched data.

Appendix 6 provides instructions on downloading and installing the R environment software and how to publish the R shiny application. Appendix 7 contains the code used in R for the project.

**Conclusion and Future Work**

The student retention toolkit using R dashboard shows that the capability exists to provide a no-cost method of displaying student retention data at a 1,600 student university. The authors discussed the project with the university’s institutional research department responsible for performing analysis of student data and provided the website address of the dashboard. As the college teaches several courses about R, a continuing of this project would enable later students enrolled in their senior projects to expand into other areas of university and student support. One of the tasks discussed early in the project but was not utilized would be to automate ingesting of data directly from the database into R using OLE DB connection manager. Pragmatic Works Task Factory is an example of a software module that would move the data manipulation such changing data types, adding or replacing data within each column with new derived values, and mapping columns. This software is used to create data warehouses which may be a goal of the project to add data from several sources to refine the analysis of student retention.

## References

1. V. Tinto, 2006-2007. *Research and practice of student retention: what next*?, College Student Retention, 8(1), 1-19.

2. R. Croc and G. Hanson, 2001. Enrollment management and student affairs. Institutional Research: Decision Support in Higher Education.

3. G. McLaughlin and R. Howard, 2004. *People, Processes, and Managing Data* (2nd. Ed.). The Association for Institutional Research, Tallahassee, FL.

4. Executive Office of the President, Office of Management and Budget (OMB). *Standards for Maintaining, Collecting, and Presenting Federal Data on Race and Ethnicity*. https://www.federalregister.gov/documents/2016/09/30/2016-23672/standards-for-maintaining-collecting-and-presenting-federal-data-on-race-and-ethnicity. Accessed November 29, 2019.

5. College Board. *Comparison of the Major Features: Old SAT and New SAT* https://collegereadiness.collegeboard.org/sat/inside-the-test/compare-old-new-specifications. Accessed November 29, 2019.

**Appendix 1. Screenshots of Student Retention Toolkit Dashboard**

Note: Some of the figures in appendix 6 may be small and hard to read. The updated Student Retention Toolkit and associated visualizations can be found at: https://mary-donahoo.shinyapps.io/Retention/

Figure 1. Splash screen with menus 

Figure 2. Student Retention by State – graphically and tabular data

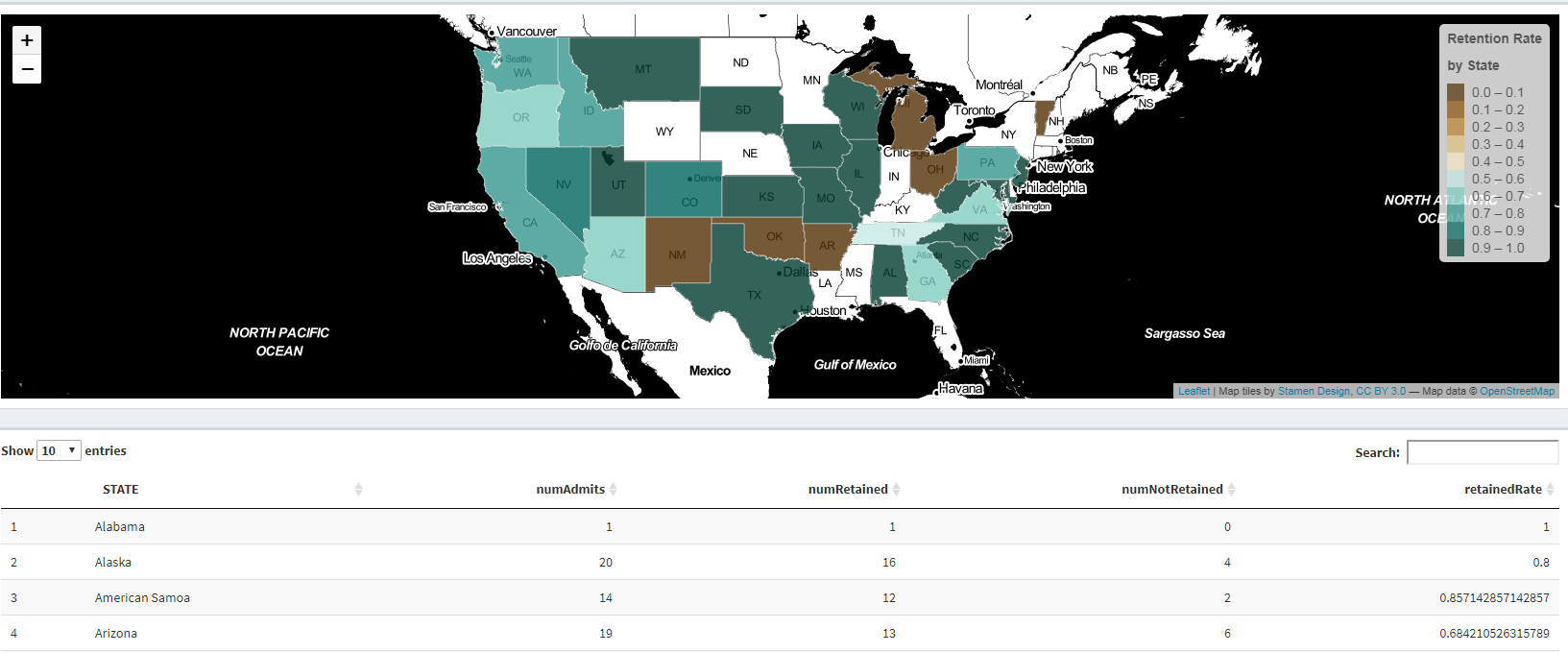


Figure 3. Student Retention by Race/Ethnicity—categorical comparisons

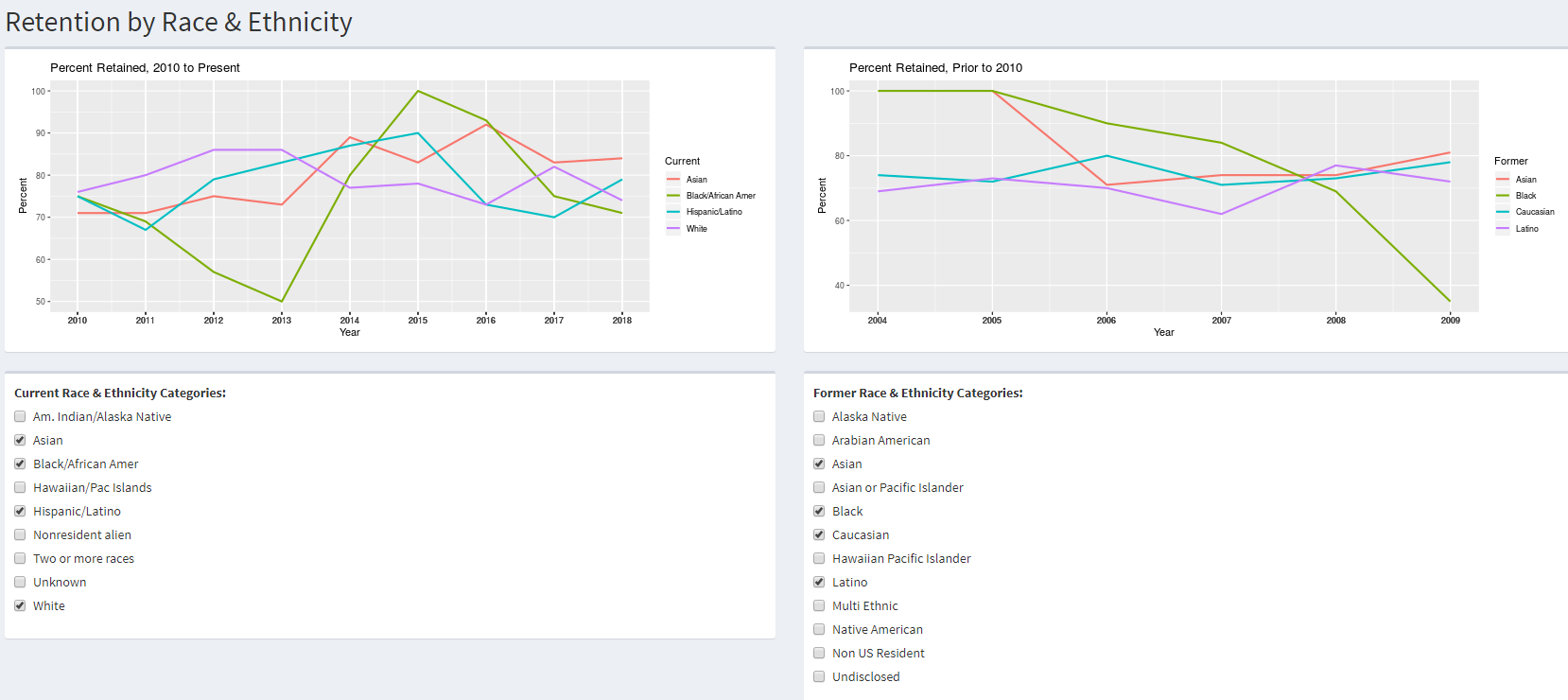


Figure 4. Retention by Gender—by number and percentage.

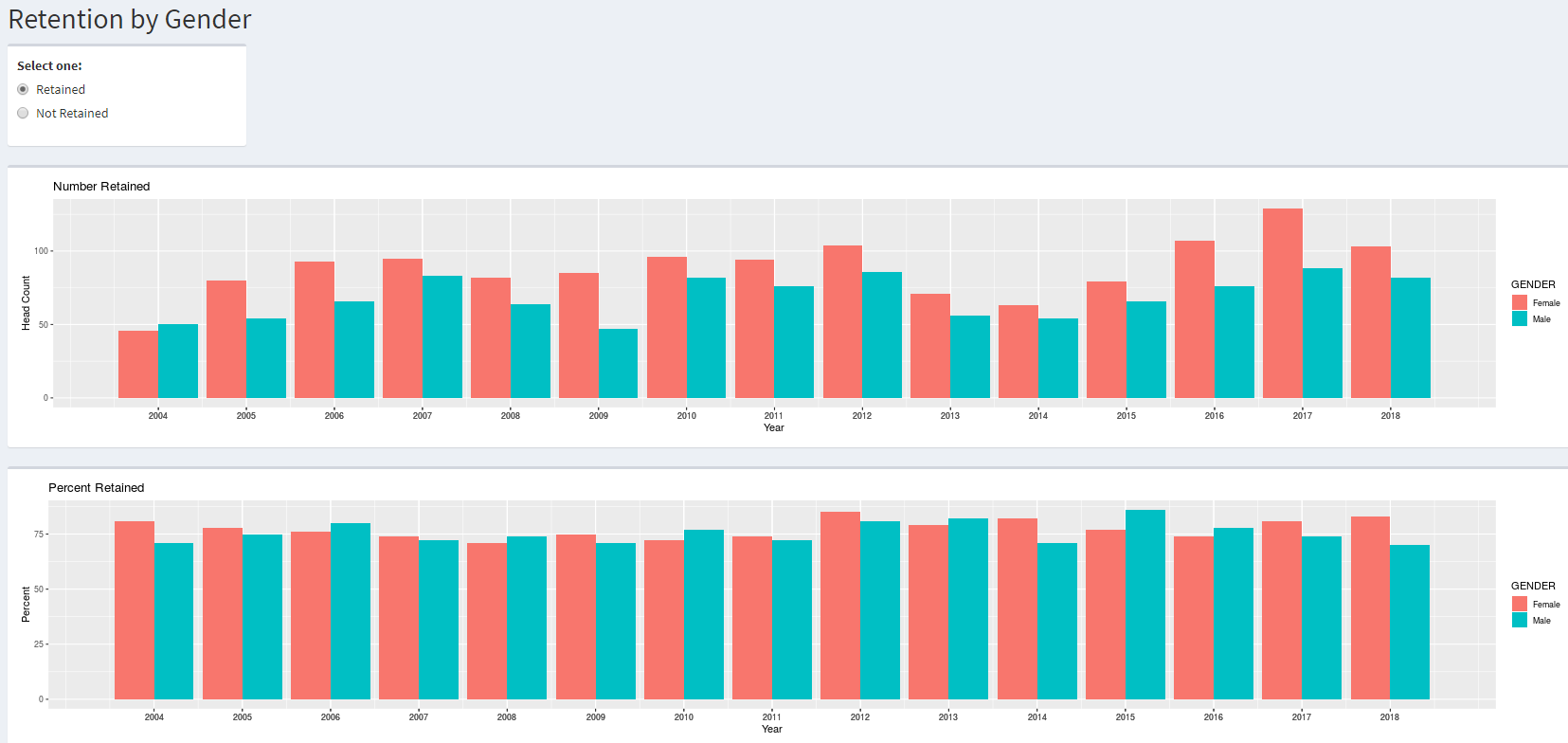


Figure 5. Retention by Major using menus and time period slider

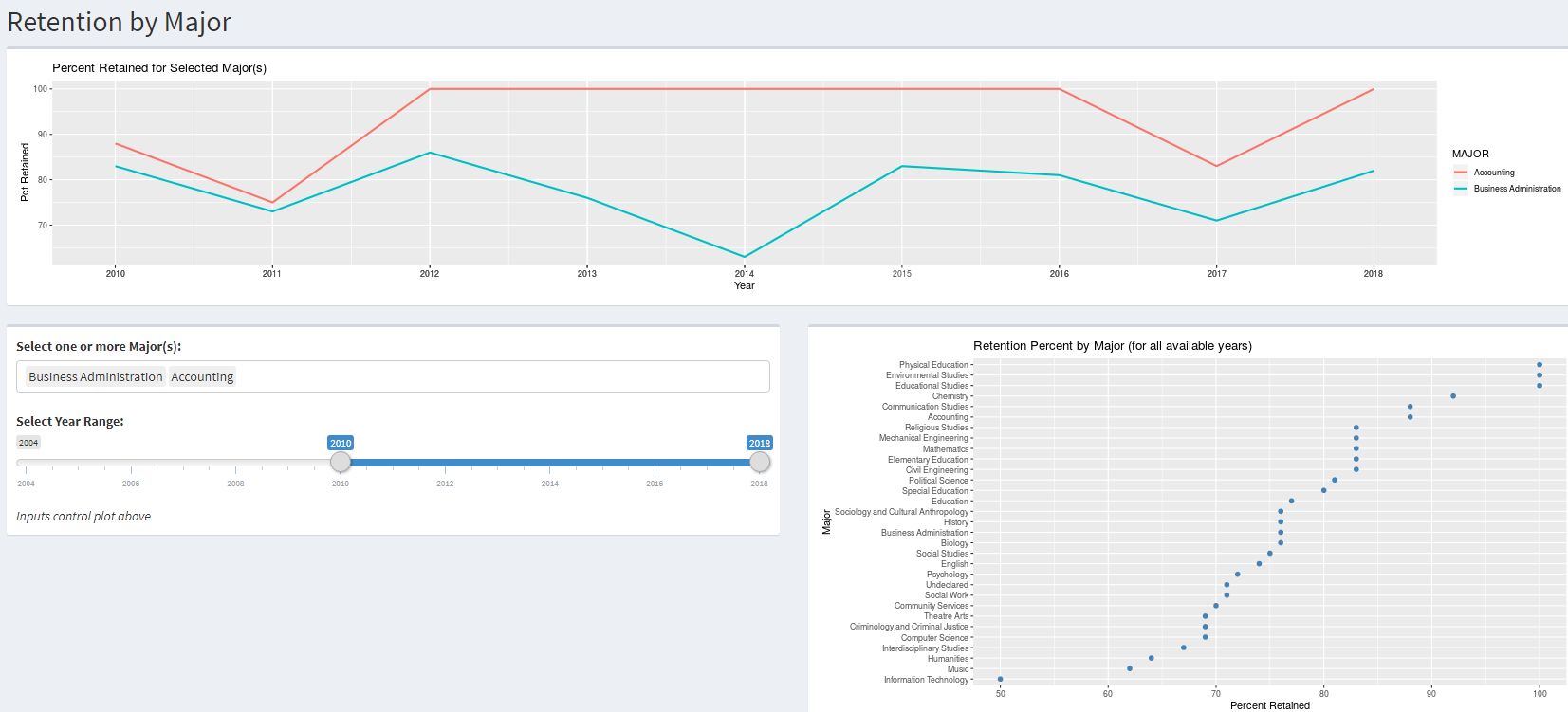
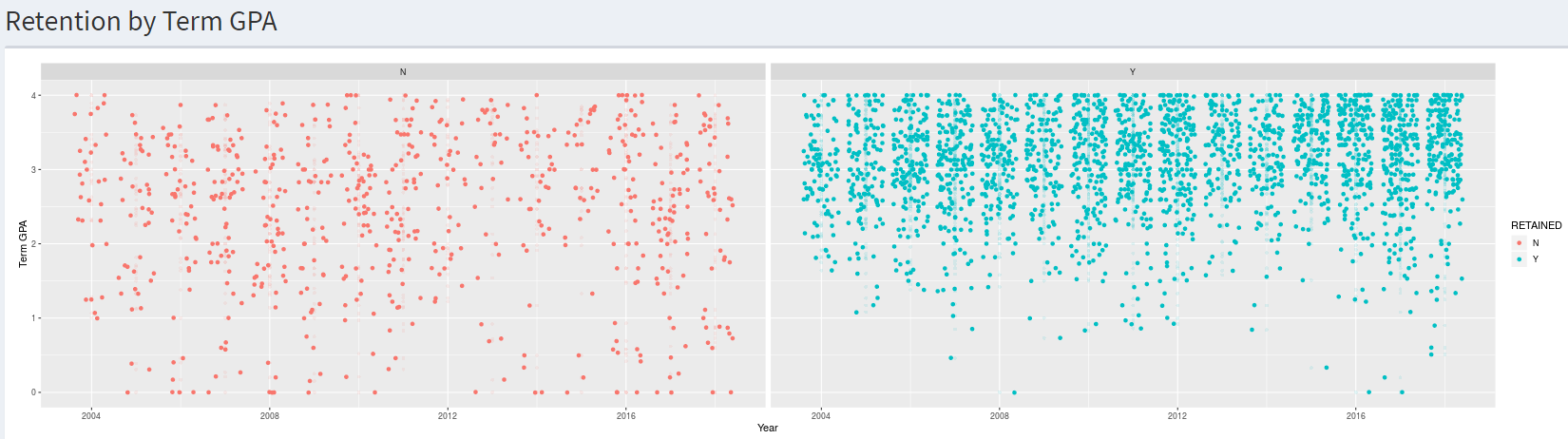


Figure 6. Retention by Academic Index



Figure 7. Retention by Term GPA



**Appendix 2. Data Dictionary**

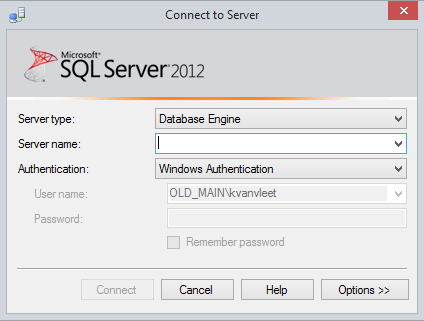
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Ref. Nbr.** | **Variable names** | **Data Type** | **Field Size** | **Example** | **Description** |
| 1 | RID | varchar | 10 | R000000123 | Unique id (external resource maps RID to student id) |
| 2 | YEAR | varchar | 4 | 2019 | Year (Calendar) |
| 3 | TERM | varchar | 10 | Fall | Term |
| 4 | COHORT | varchar | 10 | F191xFR | 1st digit is the term, 2nd and 3rd digits are the last two digits of the year, and last four digits are the college attend status |
| 5 | COHORTSTATUS | varchar | 8 | Fulltime | Full time or part time status based on enrolled credits in initial fall term |
| 6 | GENDER | varchar | 6 | Female | Gender (Male, Female, or n/a for unknown/undisclosed) |
| 7 | RE\_FORMER | varchar | 40 | Caucasian | Federal race/ethnicity categories used before the Fall of 2010 |
| 8 | RE\_CURRENT | varchar | 40 | White | Federal race/ethnicity categories used for Fall of 2010 and after |
| 9 | FIRSTGEN | varchar | 3 | Y, N, NULL | First generation |
| 10 | DEGREE | varchar | 40 | BA | Degree |
| 11 | MAJOR | varchar | 40 | Computer Science | First major |
| 12 | STEM | varchar | 1 | Y | "Y" if Computer Science, Information Technology, Civil Engineering, Mechanical Engineering, Biology, or Chemistry |
| 13 | MILSTATUS | varchar | 40 | Active Duty | Veteran/military status |
| 14 | INITIALCREDITS | numeric | 6, 2 | 15 | Enrolled credits in initial fall term |
| 15 | COUNTY | varchar | 20 | Thurston | County address at acceptance |
| 16 | STATE | varchar | 40 | Washington | State address at acceptance |
| 17 | COUNTRY | varchar | 40 | Saudi Arabia | Country address at acceptance |
| 18 | HSGPA | varchar | 4 | 0.00 - 4.00 | High school GPA |
| 19 | ACT | varchar | 4 | 1 - 36 | ACT composite score |
| 20 | FORMERSATMATH | varchar | 4 | 200 - 800 | SAT Math score taken before March 2016 |
| 21 | FORMERSATREAD | varchar | 4 | 200 - 800 | SAT Reading score taken before March 2016 |
| 22 | FORMERSATWRIT | varchar | 4 | 200 - 800 | SAT Math score taken before March 2016 |
| 23 | CURRNTSATMATH | varchar | 4 | 200 - 800 | SAT Math score taken before March 2016 |
| 24 | CURRENTSATEBRW | varchar | 4 | 200 - 800 | SAT Math score taken before March 2016 |
| 25 | ACAINDEX | varchar | 4 | 1 - 5 | Academic Index |
| 26 | PELL | varchar | 1 | Y/N | Received Pell |
| 27 | WSNG | varchar | 1 | Y/N | Received WA State Need Grant or WA College Grant (fall of 2019 and later) |
| 28 | RESCOM | varchar | 1 | Y, N, or O | "Y" for resident, "N" for Commuter, "O" for Other (changed from resident to commuter status; or study abroad) |
| 29 | RETAINED | varchar | 1 | Y/N | "Y" if student enrolled in the following year |
| 30 | RETEXCLUSION | varchar | 1 | Y/N | "Y" if student has an exclusion date prior to the end of the Fall semester in the following year |

**Appendix 3. Glossary**

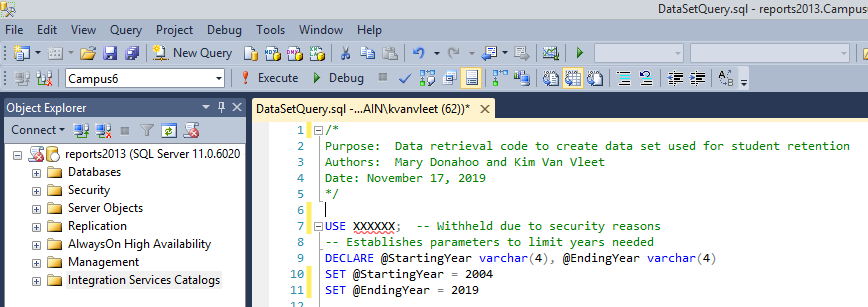
|  |  |
| --- | --- |
| Item | Description |
| UNIQUE\_ID | A unique ID derived from the student's ID to prevent disclosure |
| YEAR | Cohort Year |
| TERM | Cohort term (Fall) |
| COHORT | Code value for cohort. First position designates cohort term (F = Fall, S = Spring), second and third position designates the last two digits of the cohort year, and the fourth through seventh position is the college attend status. "1xFR" indicates a student in a "first time" cohort. Note: the "FR" means "freshman" which is a misnomer because first-time students may matriculate with a class level of other than freshman due to college credits earned before graduating from high school. Example: "F141xFR" indicates a student in the Fall or 2014 first-time cohort. |
| COHORTSTATUS | Indicates student's status at matriculation (FULLTIME or PARTTIME) |
| GENDER | Female, Male, or NULL for Undisclosed |
| RE\_FORMER | Federal race/ethnicity values prior to the Fall 2010 (Alaska Native [2004], Arabian American [to 2007], Asian, Asian or Pacific Islander [to 2006], Black, Caucasian, Hawaiian Pacific Islander, Latino, Multi Ethnic, Native American, Non US Resident, Undisclosed, and NULL) |
| RE\_CURRENT | Federal race/ethnicity categories that began in the Fall of 2010. There are two ethnicities (Hispanic or Latino and Not Hispanic and Latino), and five race categories. Non-resident aliens are categorized as such and aren't reported with an ethnicity or race. New students admitted in the Fall of 2010 were asked the new race/ethnicity category questions. Since the new categories asked the leading "Hispanic/Latino" question, the race and ethnicity selections made by returning students couldn't be converted to the new categories. Starting in the Fall of 2010, and once a term in several subsequent terms, returning students were asked to update their race/ethnicities to the new categories using SelfService keeping unknowns to a minimum (less than 8%). The former and current race/ethnicity categories are stored in separate database tables. |
|  | *Am. Indian/Alaska Native (American Indian or Alaska Native). A person having origins in any of the original peoples of North and South America (including Central America), and who maintains tribal affiliation or community attachment.* |
| RE\_CURRENT (cont) | *Asian. A person having origins in any of the original peoples of the Far East, Southeast Asia, or the Indian subcontinent including, for example, Cambodia, China, India, Japan, Korea, Malaysia, Pakistan, the Philippine Islands, Thailand, and Vietnam.* |
|  | *Black/African Amer (Black or African American). A person having origins in any of the black racial groups of Africa. Terms such as "Haitian" or "Negro" can be used in addition to "Black or African American."* |
|  | *Hispanic/Latino (Hispanic or Latino). A person of Cuban, Mexican, Puerto Rican, South or Central American, or other Spanish culture or origin, regardless of race. The term, "Spanish origin," can be used in addition to "Hispanic or Latino."* |
|  | *Hawaiian/Pac Islands (Native Hawaiian or Other Pacific Islander). A person having origins in any of the original peoples of Hawaii, Guam, Samoa, or other Pacific Islands.* |
|  | *White. A person having origins in any of the original peoples of Europe, the Middle East, or North Africa.* |
|  | *Nonresident alien* |
|  | *Two or more races* |
|  | *Unknown (Undisclosed)* |
| FIRSTGEN | First Generation is the first in their family to pursue college education. Domestic students are asked the first generation question on their application. However, students on visas are not asked the first generation question. The First Generation field in the student management system has only a "Y" for students that are first generation which assumes that all of the other students are not First Generation, Since the students on a Visa aren't asked a first generation question, "n/a" is used for these students. |
| DEGREE | The degree is the basis of the students' credential at graduation |
|  | *BA - Bachelor of Arts* |
|  | *BS - Bachelor of Science* |
|  | *BSCE - Bachelor of Science in Civil Engineering* |
|  | *BSIT - Bachelor of Science in Information Technology* |
|  | *BSME - Bachelor of Science in Mechanical Engineering* |
|  | *BSN - Bachelor of Science in Nursing* |
|  | *BSW - Bachelor of Social Work* |
|  |  |
| MAJOR | The students' major and associated degrees are determined at the end of first term and may not reflect that major (and degree) at admittance or at the start of the first term. Only the first major is included in the query and resulting dataset. Some majors have ended since 2004 and others have started since 2004. Those are annotated as such. Majors without an annotation existed since 2004. |
|  | *Accounting* |
|  | *Biology* |
|  | *Business Administration* |
|  | *Chemistry* |
|  | *Civil Engineering* |
|  | *Communication Studies (started in the Fall of 2014)* |
|  | *Community Services (taught out Fall 2015)* |
|  | *Computer Science* |
|  | *Criminology and Criminal Justice* |
|  | *Education* |
|  | *Educational Studies (started in the Fall of 2015)* |
|  | *Elementary Education* |
|  | *English (replaced by Literary Studies in the Fall of 2019)* |
|  | *Environmental Studies (srarted in the Fall of 2017)* |
|  | *Exercise Science (started in the Fall of 2019)* |
|  | *History* |
|  | *Humanities (taught out Spring 2010)* |
|  | *Information Technology (started in the Fall of 2017)* |
|  | *Interdisciplinary Studies (started in the Fall of 2009)* |
|  | *Literary Studies (started in the Fall of 2019)* |
|  | *Mathematics* |
|  | *Mechanical Engineering* |
|  | *Music* |
|  | *Nursing (taught out Fall of 2000; re-started in the Spring of 2016)* |
|  | *Physical Education* |
|  | *Political Science* |
|  | *Pre-Nursing (started in the Fall of 2019)* |
|  | *Psychology* |
|  | *Religious Studies* |
|  | *Secondary Education (started in the Fall of 2019)* |
|  | *Social Studies (taught out Spring 2010)* |
|  | *Social Work (started in the Fall of 2012)* |
|  | *Sociology and Cultural Anthropology* |
|  | *Special Education* |
| MAJOR (cont) | *Theatre Arts* |
|  | *Undeclared* |
| STEM | The following majors are categorized as STEM (Science, Technology, Engineering, and Mathematics) programs: Computer Science, Information Technology, Civil Engineering, Mechanical Engineering, Biology, Mathematics, and Chemistry. |
| MILSTATUS | Military Status. This variable denotes the students' military or veteran status or association with the miltary at the end of the students' initial term. |
|  | *Active Duty* |
|  | *Civilian* |
|  | *Disabled Veteran* |
|  | *Disabled Veteran Spouse* |
|  | *Family Member* |
|  | *National Guard* |
|  | *Reservist* |
|  | *Veteran* |
| INITIALCREDITS | Student's enrolled credits in initial term |
| COUNTY | Student's county on their permanent address record at the time of admit |
| STATE | Student's state on their permanent address record at the time of admit |
| COUNTRY | Student's country on their permanent addrss record at the time of admit |
| HSGPA | High School GPA. Office of International Programs do not enter high school GPAs. |
| ACT | ACT Composite score |
| FORMERSATMATH | SAT math score taken prior to March 2016 |
| FORMERSATREAD | SAT reading score taken prior to March of 2016 |
| FORMERSATWRIT | SAT writing score taken prior to March of 2016 |
| CURRENTSATMATH | SAT math score taken March 2016 or later |
| CURRENTSATEBRW | SAT Evidence-Based Reading Writing score taken March 2016 or later |
| ACAINDEX | Academic Index. This number between 1 and 5, inclusive, is assigned to domestic students for the purpose of awarding scholarships |
| PELL | Pell Grant received in the student's initial fall term. If student received any amount then "Y". Otherwise "N" |
| WSNG | WA State Need Grant (changed to WA College Grant in the Fall of 2019) received in the student's initial fall term. If student received any amount, then "Y", otherwise "N". |
| RESCOM | Resident, Commuter status. This is the students' residency in their term. "Other" is usually used when a student starts the term living in the resident hall but changes to commuter during the semester. |
| TERMGPA | Student's initial term GPA. Students who withdrew or have no Term GPA will have a Term GPA of NULL. |
| WITHDREW | Flag students who withdrew in their initial term -- "Y" if yes (received "W" final grades on all classes) |
| RETAINED | Retained flag of "Y" if student was enrolled in credits in the following fall term |
| RETEXCLUSION | Retention exclusion. If a student is flagged as excluded from the retention rate calculations ("Y") if they left the institution prior to the end of the fall term in the following year for one of the following reasons: death or total and permanent disability; service in the armed forces (including those called to active duty); service with foreign aid service of the federal government, such as the Peace Corps; or service on official church missions. |

**Appendix 4. Retrieving Data Set from SQL Server.**

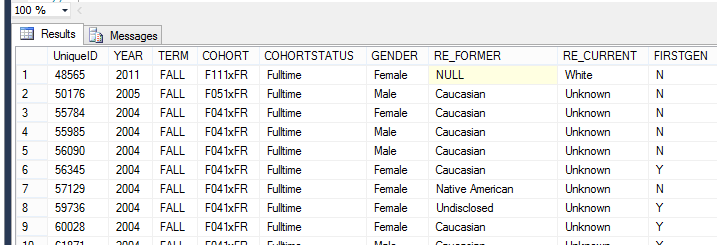
1. Open SQL Server Management Studio (SSMS). The “Connect to Server” window appears.
2. Connect to a database instance. To protect the security of the database and prevent unauthorized intrusions, the server and user names have been withheld and can be obtained from the authors.
3. Click “Connect”.



1. In Windows Explorer, double-click on the query file. The file should open in SQL Server Management Studio.
2. Add the desired cohort starting year in the “SET @StartingYear = \_\_\_\_” statement (line 10) and the required cohort ending year in the “SET @EndingYear = \_\_\_\_” statement (line 11)
3. In SSMS, click “Execute”. The query will point to the database specified in the “USE” statement (withhold).



1. Click anywhere inside the Results window.



1. Right-click and then select “Select All”. Right-click again and select “Copy with Headers”.
2. Open a spreadsheet and paste the contents.
3. Save the spreadsheet as an Excel file (.xlxs) with a filename of   
   “Retention\_withXref.xlsx” (without the quotes) into the R data folder.

**Appendix 5. SQL Query to retrieve student data set.**

/\*

Purpose: Data retrieval code to create data set used for student retention

Authors: Mary Donahoo and Kim Van Vleet

Date: December 3, 2019

\*/

-- Establishes parameters to limit years needed

DECLARE @StartingYear varchar(4), @EndingYear varchar(4)

SET @StartingYear = 2004 -- no earlier than 2004 due to incomplete data

SET @EndingYear = 2018 -- last cohort year that retention can be derived

SELECT CONVERT(int,A.people\_id) + XXXX AS [UniqueID] -- constant is withheld

,A.Academic\_year AS [YEAR]

,A.Academic\_term AS [TERM]

-- code for students cohort

,SC.Cohort\_id AS [COHORT]

-- Indicates whether students was fulltime or parttime in initial term

,CASE WHEN SC.Sub\_Cohort\_id = 'FULLTIME' THEN 'Fulltime'

WHEN SC.Sub\_Cohort\_id = 'PARTTIME' THEN 'Parttime'

ELSE NULL END AS [COHORTSTATUS]

-- Gender: All binary now but it may not be the case in the future

,CASE WHEN D.Gender = 'F' THEN 'Female'

WHEN D.Gender = 'M' THEN 'Male'

ELSE COALESCE(D.Gender, NULL) END AS [GENDER]

-- Old race/ethnicity categories

,COALESCE(CE.Long\_desc, NULL) AS [RE\_FORMER]

-- New race/ethnicity categories

,CASE WHEN IE.Ethnicity = 'AmericanIndianorAlaskaNative' THEN 'Am. Indian/Alaska Native'

WHEN IE.Ethnicity = 'BlackorAfricanAmerican' THEN 'Black/African Amer'

WHEN IE.Ethnicity = 'Hispanic' THEN 'Hispanic/Latino'

WHEN IE.Ethnicity = 'Asian' THEN 'Asian'

WHEN IE.Ethnicity = 'MultiEthnic' THEN 'Two or more races'

WHEN IE.Ethnicity = 'NativeHawaiianorOtherPacificIslander' THEN 'Hawaiian/Pac Islands'

WHEN IE.Ethnicity = 'NonResidentAlien' THEN 'Nonresident alien'

WHEN IE.Ethnicity = 'White' THEN 'White'

WHEN IE.Ethnicity IS NULL THEN 'Unknown' END AS [RE\_CURRENT]

-- First generation; can be "Y" or "y" or NULL (data not entered) or 'Blank' (data entered but then deleted)

,CASE WHEN UD.First\_generation IN ('Y','y') THEN 'Y'

-- International Office doesn't ask for first generation so incorrectly deafult to "no" instead of "n/a"

WHEN IE.Ethnicity = 'NonResidentAlien' AND D.Visa IS NOT NULL THEN NULL

WHEN UD.First\_generation = 'N' THEN 'N'

ELSE 'N' END AS [FIRSTGEN]

,A.Degree AS [DEGREE] -- at end of first fall term

,CC.Long\_Desc AS [MAJOR] -- at end of first fall term

-- STEM program

,CASE WHEN A.Degree = 'BS' OR A.curriculum IN (

'110701', -- CSC

'110103', -- BSIT

'140801', -- BSCE

'141901', -- BSME

'143501', -- BSIE (Proposed)

'260101', -- BIO

'270101', -- MTH

'400501') -- CHM

THEN 'Y' ELSE 'N' END AS [STEM]

,CV.Long\_Desc AS [MILSTATUS]

,A.Credits AS [INITIALCREDITS]

-- Location at accept

,COALESCE(LAA.County, NULL) AS [COUNTY]

,COALESCE(LAA.[State], NULL) AS [STATE]

,COALESCE(LAA.Country, NULL) AS [COUNTRY]

-- HS GPA

,CASE WHEN HS.Gradepoint\_average IS NULL THEN NULL

WHEN HS.Gradepoint\_average = 0 THEN NULL

ELSE CONVERT(varchar, HS.Gradepoint\_Average) END AS [HSGPA]

-- ACT Scores

,COALESCE(CONVERT(varchar, T1.ACT\_composite), NULL) AS [ACT]

,COALESCE(CONVERT(varchar, T1.SAT\_Math), NULL) AS [FORMERSATMATH]

,COALESCE(CONVERT(varchar, T1.SAT\_Reading), NULL) AS [FORMERSATREAD]

,COALESCE(CONVERT(varchar, T1.SAT\_Writing), NULL) AS [FORMERSATWRIT]

-- New SAT

,COALESCE(CONVERT(varchar, SAT3.Math), NULL) AS [CURRENTSATMATH]

,COALESCE(CONVERT(varchar, SAT3.Readw), NULL) AS [CURRENTSATEBRW]

-- Academic Index

,CASE WHEN SC.Cohort\_id LIKE 'F0%1xFR' THEN NULL

ELSE COALESCE(UD.Aca\_index, NULL) END AS [ACAINDEX]

-- Pell

,CASE WHEN [PELL Amount] > 0 THEN 'Y'

ELSE 'N' END AS [PELL]

-- WSNG

,CASE WHEN [WSNG Amount] > 0 THEN 'Y'

ELSE 'N' END AS [WSNG]

-- Resident/Commuter

,CASE WHEN R.Resident\_commuter = 'R' THEN 'Resident'

WHEN R.Resident\_commuter = 'C' THEN 'Commuter'

WHEN R.Resident\_Commuter = 'O' THEN 'Other'

ELSE NULL END AS [RESCOM]

-- Term GPA

,CASE WHEN TG.GPA < 0.001 AND EXISTS

(SELECT \* FROM TRANSCRIPTDETAIL TD

WHERE TD.PEOPLE\_CODE\_ID = TG.PEOPLE\_CODE\_ID

AND TD.ACADEMIC\_YEAR = TG.ACADEMIC\_YEAR

AND TD.ACADEMIC\_TERM = TG.ACADEMIC\_TERM

AND TD.ADD\_DROP\_WAIT = 'A'

AND TD.FINAL\_GRADE IN ('F','XF'))

THEN '0.00'

WHEN TG.GPA < 0.001 -- F/XF not exists is implied

THEN NULL -- change "WD" to NULL

ELSE CONVERT(varchar,TG.GPA) END AS [TermGPA]

-- flag for withdrawn students

,CASE WHEN TG.GPA < .001 AND EXISTS

(SELECT \* FROM TRANSCRIPTDETAIL TD

WHERE TD.PEOPLE\_CODE\_ID = TG.PEOPLE\_CODE\_ID

AND TD.ACADEMIC\_YEAR = TG.ACADEMIC\_YEAR

AND TD.ACADEMIC\_TERM = TG.ACADEMIC\_TERM

AND TD.ADD\_DROP\_WAIT = 'A'

AND TD.FINAL\_GRADE IN ('F','XF'))

THEN NULL

WHEN TG.GPA < 0.001 -- F/XF not exists is implied

THEN 'Y' END AS [WITHDREW]

-- Looks for academic record in following year for retained students

,CASE WHEN A2.[NEXT\_YEAR\_CREDITS] > 0.00 THEN 'Y'

ELSE 'N' END AS [RETAINED]

-- If student has an exclusion date by the end of the fall term in the following year

,CASE WHEN SC.Exclusion\_Date < AC.End\_date THEN 'Y'

ELSE 'N' END AS [RETEXCLUSION]

FROM Academic A

-- Table includes cohort code and fulltime/partime status

LEFT JOIN StudentCohort SC

ON A.People\_code\_id = SC.People\_code\_id

-- gender, military status

LEFT JOIN Demographics D

ON A.People\_code\_id = D.People\_code\_id

AND A.Academic\_year = D.Academic\_year

AND A.Academic\_term = D.Academic\_term

AND D.Academic\_Session = ''

-- old race/ethnicities prior to Fall of 2010

LEFT JOIN Code\_ethnicity CE

ON D.Ethnicity = CE.Code\_value

-- new race/ethnicities Fall of 2010 and later

LEFT JOIN smu.vw\_IPEDSReportingEthnicity IE

ON A.People\_code\_id = IE.People\_code\_id

-- Converts academic curriculum as code value to plain language major

LEFT JOIN Code\_curriculum CC

ON A.curriculum = CC.Code\_value

-- Converts military status to plain language

LEFT JOIN Code\_veteran CV

ON D.Veteran = CV.Code\_value

-- First generation and academic index

LEFT JOIN UserDefinedInd UD

ON A.People\_code\_id = UD.People\_code\_id

-- Address when student was admitted

LEFT JOIN smu.vw\_IR\_LocationAtAccept LAA

ON A.People\_code\_id = LAA.People\_code\_id

-- High school GPA

LEFT JOIN smu.vw\_ADM\_HighSchoolInfo HS

ON A.people\_code\_id = HS.people\_code\_id

-- View with ACT and old SAT (prior to March of 2016) scores

LEFT JOIN smu.vw\_ADM\_TestScores T1

ON A.People\_code\_id = T1. People\_code\_ID

-- new SAT scores (March 2016 and later)

LEFT JOIN

(SELECT A.people\_code\_id,MAX(M.raw\_score) AS [Math],

MAX(R.raw\_score) AS [Readw]

FROM Academic A

LEFT JOIN dbo.TestScores M

ON A.people\_code\_id = M.people\_code\_id

AND M.test\_id = 'SAT3'

AND M.test\_type = 'MATH'

AND M.Raw\_score between 200 AND 800

LEFT JOIN dbo.TestScores R

ON A.people\_code\_id = R.people\_code\_id

AND R.test\_id = 'SAT3'

AND R.test\_type = 'READWRIT'

AND R.Raw\_score between 200 AND 800

WHERE M.people\_code\_id IS NOT NULL --LIKE '%'

OR R.people\_code\_id IS NOT NULL -- LIKE '%'

GROUP BY A.people\_code\_id

) SAT3

ON A.people\_code\_id = SAT3.people\_code\_id

-- Pell Grant

LEFT JOIN

(SELECT People\_org\_code\_id, Academic\_year

,SUM(Amount) AS [Pell Amount]

FROM ChargeCredit

WHERE Charge\_credit\_code = 'AGRTAPELL'

GROUP BY People\_org\_code\_id, academic\_year

) PELL

ON A.People\_code\_id = PELL.People\_org\_code\_id

AND SUBSTRING(SC.Cohort\_id, 1, 1) = 'F'

AND SUBSTRING(SC.Cohort\_id, 2, 2)

= SUBSTRING(PELL.Academic\_year, 3, 2)

-- Washington State Need Grant (changed to Washington College Grant in Fall 2019)

LEFT JOIN

(SELECT People\_org\_code\_id, Academic\_year

,SUM(Amount) AS [WSNG Amount]

FROM ChargeCredit

WHERE Charge\_credit\_code IN ('AGRTAWSNG','PCHKKWSNG','AGRTAWACOL','PCHKAWSNG')

GROUP BY People\_org\_code\_id, academic\_year

) WSNG

ON A.People\_code\_id = WSNG.People\_org\_code\_id

AND SUBSTRING(SC.Cohort\_id, 1, 1) = 'F'

AND SUBSTRING(SC.Cohort\_id, 2, 2) = SUBSTRING(WSNG.Academic\_year, 3, 2)

-- Resident (dorms), commuters

LEFT JOIN Residency R

ON A.People\_code\_id = R.People\_code\_id

AND A.Academic\_year = R.Academic\_year

AND A.Academic\_term = R.Academic\_term

AND A.Academic\_session = R.Academic\_session

-- First term GPA

LEFT JOIN TranscriptGPA TG

ON TG.PEOPLE\_CODE\_ID = A.PEOPLE\_CODE\_ID

AND TG.ACADEMIC\_YEAR = A.ACADEMIC\_YEAR

AND TG.ACADEMIC\_TERM = A.ACADEMIC\_TERM

AND TG.ACADEMIC\_SESSION = A.ACADEMIC\_SESSION

AND TG.RECORD\_TYPE = 'O'

AND TG.PROGRAM = ''

-- Determines if student was enrolled in the following year

LEFT JOIN

(SELECT PEOPLE\_CODE\_id, ACADEMIC\_YEAR, ACADEMIC\_TERM, ACADEMIC\_SESSION

, SUM(CREDITS) AS [NEXT\_YEAR\_CREDITS]

FROM ACADEMIC

GROUP BY PEOPLE\_CODE\_id, ACADEMIC\_YEAR, ACADEMIC\_TERM, ACADEMIC\_SESSION

) A2

ON A2.PEOPLE\_CODE\_ID = A.PEOPLE\_CODE\_ID

AND CONVERT(int,A2.Academic\_year) = CONVERT(int,A.Academic\_year) + 1

AND A2.ACADEMIC\_TERM = A.ACADEMIC\_TERM

AND A2.ACADEMIC\_SESSION = A.ACADEMIC\_SESSION

-- Used to determine if student's exclusion date falls before end of term in following year

LEFT JOIN AcademicCalendar AC

ON A.Academic\_year + 1 = AC.Academic\_year

AND A.Academic\_term = AC.Academic\_term

AND AC.Academic\_Session = '00'

WHERE A.Program = 'UG'

-- look for the last two years from the cohort\_ID

AND A.Academic\_year = CONCAT('20',SUBSTRING(SC.Cohort\_id, 2, 2))

AND A.Academic\_Term = 'Fall'

AND A.Academic\_Session = ''

AND A.[status] = 'A' -- Excludes inactivated records (i.e., old majors after a change of major)

AND A.Primary\_Flag = 'Y' -- Includes only students' first major and excludes their second major, dual major, minor, concentration, and any inactivated record

AND A.Academic\_flag = 'Y' -- Includes records where application was not canceled

-- use only cohorts starting in 2004; flagging of all students in 2000 to 2003 cohorts has not been done completely

AND (SC.Cohort\_id IN ('F041xFR','F051xFR','F061xFR','F071xFR','F081xFR','F091xFR')

OR SC.Cohort\_id LIKE 'F1%1xFR'

OR SC.Cohort\_id LIKE 'F2%1xFR') -- ready to handle Fall 2020 cohorts and beyond

AND A.Academic\_year BETWEEN @StartingYear AND @EndingYear

**Appendix 6. Software Instructions for R.**

To develop an R application, the software components of R, RTools and RStudio software can be downloaded for free. The R and RTools software can be found in the following web address:

https://cran.r-project.org/bin/windows/

To install R from the R for Windows page, click on the link “Install R for the first time” and follow the instructions. After R has been installed, return to the R for Windows page and click on the RTools link and follow the instructions for downloading and installing Rtools. Once R and RTools have been installed, download and install RStudio found at this web address:

https://rstudio.com/products/rstudio/download/

As an alternative to installing the R, RTools and RStudio software on your computer, a cloud version of RStudio is available at this web address:

https://rstudio.cloud/

To publish an R Shiny application, a shinyapps.io account is required and is available at this web address:

https://www.shinyapps.io/.

The shinyapps.io account is free to use for up to five applications and 25 active hours per month. Paid plans are also available starting at $9 per month for 25 applications and 100 active hours.

**Appendix 7. R code used in the project.**

#

# This is a Shiny web application. You can run the application by clicking

# the 'Run App' button above.

#

# Find out more about building applications with Shiny here:

#

# http://shiny.rstudio.com/

# https://rstudio.github.io/shinydashboard/structure.html

library(shiny)

library(shinydashboard)

library(dplyr)

library(leaflet)

library(rgdal)

library(DT)

library(tidyverse)

library(readxl)

retention <- read\_excel("data/Retention\_withXref.xlsx")

DataDictionary <- read\_excel("data/DataDictionary.xlsx")

states <- readOGR("data/cb\_2018\_us\_state\_500k/cb\_2018\_us\_state\_500k.shp")

us <- retention %>%

mutate(retained = ifelse(RETAINED == 'Y',1,0))

us <- subset(us,is.element(us$STATE, states$NAME))

# is.element(us$STATE, states$NAME)

states <- subset(states,is.element(states$NAME, us$STATE))

# is.element(states$NAME, us$STATE)

us <- us[order(match(us$STATE, states$NAME)),]

#create bins based on the retained rate in us

bins <- c(0.0,0.1,0.2,0.3,0.4,0.5,0.6,0.7,0.8,0.9,1.0)

#give pal a number, it returns a color, see RColorBrewer package "RdYlBu"

pal <- colorBin("BrBG", domain = c(0,1), bins = bins)

dataGender <- retention %>%

group\_by(GENDER,YEAR,RETAINED) %>%

summarize(N = n()) %>%

mutate(freq = N/sum(N),

pct = round((freq\*100),0)) %>%

na.omit()

r4 <- retention %>%

filter (YEAR >= 2004) %>%

filter (YEAR <= 2018) %>%

group\_by(MAJOR, RETAINED) %>%

summarize(N = n()) %>%

mutate(freq = N/sum(N),

pct = round((freq\*100),0)) %>%

na.omit() %>%

filter (RETAINED == 'Y')

# Get unique list of majors for Select list on Major tab

list\_of\_majors <- sort(unique(retention$MAJOR))

dataMajor <- retention %>%

group\_by(MAJOR, YEAR, RETAINED) %>%

summarize(N = n()) %>%

mutate(freq = N/sum(N),

pct = round((freq\*100),0)) %>%

na.omit() %>%

filter (RETAINED == 'Y')

dataAcaindex <- retention %>%

filter(ACAINDEX != "n/a") %>%

group\_by(ACAINDEX,YEAR,RETAINED) %>%

summarize(N = n()) %>%

mutate(freq = N/sum(N),

pct = round((freq\*100),0)) %>%

na.omit()

dataGPA <- retention %>%

filter(TermGPA != 'WD') %>%

mutate(gpa = as.numeric(TermGPA))

# UI

ui <- dashboardPage(

skin = "red",

dashboardHeader(title = "Retention Toolkit"),

dashboardSidebar(

# sliderInput("date\_range", label = "Date Range",

# min = min(us$YEAR),

# max = max(us$YEAR),

# value = c(min(us$YEAR), max(us$YEAR)),

# sep = "",

# step = 1),

sidebarMenu(

menuItem("Dashboard", tabName = "dashboard", icon = icon("dashboard")),

menuItem("Charts",icon = icon("line-chart"),

menuSubItem("State at Admit", tabName = "stateAtAdmit", icon = icon("map")),

menuSubItem("Race & Ethnicity", tabName = "raceEthnicity",icon = icon("pie-chart")),

menuSubItem("Gender", tabName = "gender", icon = icon("bar-chart")),

menuSubItem("Major", tabName = "major",icon = icon("cog",lib="glyphicon")),

menuSubItem("AcaIndex", tabName = "acaIndex",icon = icon("heart",lib="font-awesome")),

menuSubItem("GPA", tabName = "gpa",icon = icon("crown"))

),

menuItem("Data Dictionary", tabName = "datadict", icon = icon("book")),

menuItem("Raw Data", tabName = "rawdata", icon = icon("database")),

menuItem("Help", tabName = "helpdoc", icon = icon("flag"))

)

),

dashboardBody(

tabItems(

tabItem(tabName = "dashboard",

tags$img(style = "height:15%; width:90%;", src = "SMU\_ColorLogo\_CMYK.png"),

HTML("<h1><center>Student Retention Toolkit</center></h1>"),

HTML("<h3><center>by Kim Van Vleet and Mary Donahoo</center></h3>")

),

tabItem(tabName = "stateAtAdmit",

fluidRow(box(width = 12,leafletOutput(outputId = "mymap"))),

fluidRow(box(width = 12, dataTableOutput(outputId = "summary\_table" )))

),

tabItem(tabName = "raceEthnicity",

h2("Retention by Race & Ethnicity"),

fluidRow(box(width = 6,"Race & Ethnicity 2010 to present"),

box(width = 6,"Race & Ethnicity before 2010"))

),

tabItem(tabName = "gender",

h2("Retention by Gender"),

fluidRow(box(width = 2,

radioButtons(inputId = "gender\_rb",

label = "Select one: ",

c("Retained" = "Y",

"Not Retained" = "N")))),

fluidRow(box(width = 12,

plotOutput(outputId = "genderCount\_plot",height = 275))),

fluidRow(box(width = 12,

plotOutput(outputId = "genderPct\_plot", height = 275)))

),

tabItem(tabName = "major",

h2("Retention by Major"),

fluidRow(box(width = 12,

plotOutput(outputId = "major\_plot3", height = 250))),

fluidRow(box(width = 6,"These inputs control the plot above",br(),br(),

selectInput(inputId = "selected\_major",

label = "Select one or more Majors: ",

choices = list\_of\_majors,

selected = "Computer Science",

multiple = TRUE,

selectize = TRUE

),

sliderInput(inputId = "major\_year\_range",label = "Select Year Range:",

min = 2004,max = 2018,

value = c(2010,2018),

sep="",

step = 1)

),

box(width = 6,plotOutput(outputId = "major\_plot", height = 250))

)

),

tabItem(tabName = "acaIndex",

h2("Retention by Academic Index"),

fluidRow(box(width = 12,

plotOutput(outputId = "acaindexCount\_plot",height = 275))),

fluidRow(box(width = 12,

plotOutput(outputId = "acaindexPct\_plot", height = 275))),

h4("Academic Index is a number between 1 and 5 assigned to domestic students for the purpose of awarding scholarships.")

),

tabItem(tabName = "gpa",

h2("Retention by Term GPA"),

fluidRow(box(width = 12,

plotOutput(outputId = "gpa\_plot",height = 400)))

),

tabItem(tabName = "datadict",

fluidRow(box(width = 12, dataTableOutput(outputId = "data\_dict" )))

),

tabItem(tabName = "rawdata",

fluidRow(box(width = 12, dataTableOutput(outputId = "raw\_data" )))

),

tabItem(tabName = "helpdoc",

fluidPage(box(width = 12,uiOutput( outputId = "help\_pdf")))

)

)

)

)

# Server 15:00

server <- function(input, output) {

data\_input <- reactive({

us %>%

filter(YEAR >= 2004) %>%

filter(YEAR <= 2018) %>%

group\_by(STATE) %>%

summarize(numAdmits = n(),

numRetained = sum(retained)) %>%

mutate(numNotRetained = numAdmits - numRetained,

retainedRate = numRetained/numAdmits)

})

data\_input\_ordered <- reactive({

data\_input()[order(match(data\_input()$STATE, states$NAME)),]

})

labels <- reactive({

paste("<p>",data\_input\_ordered()$STATE, "</p>",

"<p>","Retained Rate: ", round(data\_input\_ordered()$retainedRate,digits = 3),

"</p>",sep="")

})

# output$mymap <- renderLeaflet(

output$mymap <- renderLeaflet(

leaflet() %>%

setView(-96,37.8,4) %>%

addProviderTiles(providers$Stamen.Toner) %>%

addPolygons(data = states,

weight = 1,

smoothFactor = 0.5,

color = "white",

fillOpacity = 0.8,

fillColor = pal(data\_input\_ordered()$retainedRate),

label = lapply(labels(),HTML)) %>%

addLegend(pal = pal,

values = data\_input\_ordered()$retainedRate,

opacity = 0.7,

position = "topright",

title = "<p>Retention Rate</p><p>by State</p>")

)

output$summary\_table <- renderDataTable(data\_input())

output$genderCount\_plot <- renderPlot({

selected\_dataGender <- dataGender %>%

filter(RETAINED == input$gender\_rb)

genderPlotTitle <- if(input$gender\_rb == "Y") "Retained" else "Not Retained"

ggplot(selected\_dataGender, aes(x = YEAR, y = N, color = GENDER)) +

geom\_line(size = 1) +

scale\_x\_continuous("Year", breaks = selected\_dataGender$YEAR) +

labs(x="Year",y="Head Count") +

ggtitle(paste("Number",genderPlotTitle))})

output$genderPct\_plot <- renderPlot({

selected\_dataGender <- dataGender %>%

filter(RETAINED == input$gender\_rb)

genderPlotTitle <- if(input$gender\_rb == "Y") "Retained" else "Not Retained"

ggplot(selected\_dataGender, aes(x = YEAR, y = pct, color = GENDER)) +

geom\_line(size = 1) +

scale\_x\_continuous("Year", breaks = selected\_dataGender$YEAR) +

labs(x="Year",y="Percent") +

ggtitle(paste("Percent",genderPlotTitle))})

output$major\_plot <- renderPlot(ggplot(r4, aes(x = pct, y = reorder(MAJOR,pct))) +

geom\_point(color = "steel blue", size = 2) +

labs(x="Percent Retained",y="Major"))

output$major\_plot3 <- renderPlot({

selected\_dataMajor <- dataMajor %>%

filter(MAJOR %in% input$selected\_major) %>%

filter(YEAR >= input$major\_year\_range[1]) %>%

filter(YEAR <= input$major\_year\_range[2])

ggplot(selected\_dataMajor, aes(x = YEAR, y = pct,color = MAJOR )) +

geom\_line(size = 1) +

scale\_x\_continuous("Year", breaks = selected\_dataMajor$YEAR) +

labs(x="Year",y="Pct Retained") +

ggtitle(paste("Percent Retained:",input$selected\_major))})

output$acaindexCount\_plot <- renderPlot(ggplot(dataAcaindex, aes(x = YEAR, y=N)) +

geom\_col(aes(fill = RETAINED)) +

facet\_wrap(~ACAINDEX, ncol = 5) +

labs(x="Year",y="Count") +

ggtitle("Retention Count by Academic Index"))

output$acaindexPct\_plot <- renderPlot(ggplot(dataAcaindex, aes(x = YEAR, y=pct)) +

geom\_col(aes(fill = RETAINED)) +

facet\_wrap(~ACAINDEX, ncol = 5) +

labs(x="Year",y="Percent") +

ggtitle("Retention Percent by Academic Index"))

output$gpa\_plot <- renderPlot(ggplot(dataGPA, aes(x = YEAR, y=gpa,color = RETAINED)) +

geom\_point(alpha = 0.1, shape = 16) +

geom\_jitter() +

facet\_wrap(~RETAINED) +

labs(x="Year",y="Term GPA"))

output$data\_dict <- renderDataTable(DataDictionary,options = list(pageLength = 15))

output$raw\_data <- renderDataTable(retention)

output$help\_pdf <- renderUI(tags$iframe(style = "height:800px; width:100%; scrolling = yes;",src = "help.pdf"))

}

# Run the application

shinyApp(ui = ui, server = server)