# Query Builder for Tables: A "No Tears" Way for Newcomers to Relate to the Relational Model

Robert H. Carver, Professor of Business Administration Stonehill College, Easton MA 02357, (508-565-1130), [rcarver@stonehill.edu](mailto:rcarver@stonehill.edu) and Senior Lecturer, Brandeis University International Business School, rcarver@brandeis.edu

# ABstract

# Some of the most creative applications in business analytics combine data from multiple sources in novel ways. When a project draws upon data from several sources, the data management tasks can be daunting for a student or a relatively new analyst. Even when you have good statistical skills, it can be challenging to brush up on SQL or other tools needed to join, subset, or otherwise wrangle data in preparation for analysis. JMP12 introduced Query Builder for querying databases, and JMP 13 went a step further adding Query Builder to the Tables menu. Even if your formal training with relational databases is non-existent or quite old, Query Builder for Tables makes it painless to perform database operations on JMP Data Tables. If you've had mixed success JOINing tables in the past, Query Builder will be a welcome addition to your repertoire. This talk presents an illustrative example that should be quite easy to follow for all attendees.

# Introduction

JMP users understand that data preparation and data wrangling are time-consuming phases of any analytics project, but college-level courses and textbooks have traditionally devoted little attention to gathering, assembling, and preparing data for analysis. Those of us who teach undergraduate courses may emphasize descriptive and inferential procedures, using neat, complete datasets to illustrate.

Fortunately, the Statistics Education community has recognized the disconnect between practice and the content of our courses, and things are moving. Gould’s influential 2010 paper laid out some of the key developments relating to big data and the Internet of Things. Similarly, Horton *et al.* 2013) and Carver and Stephens (2015) recommend expansions in the content of college courses to equip students to work with data. The recently revised GAISE College Report (2016) presents a vision for college courses that use real data to seek answers to important real-world questions, that emphasize statistical concepts over the mechanics of procedure, and that leverage available technology to facilitate learning. The GAISE report also stresses the importance of teaching statistics as part of a larger investigative process and introducing multivariate thinking early.

At the same time, students are building statistical foundations in their K-12 years. This is reflected in the growth in students sitting for the Advanced Placement Exam in statistics. In 2016, over 206,000 high school students took the exam with close to 75,000 earning college credit for introductory statistics (see Figure 1). Even larger numbers studied statistics in high school, but did not take the exam. This growth in K-12 should mean that we can dial back some traditional first-course content and make room for topics related to data preparation.

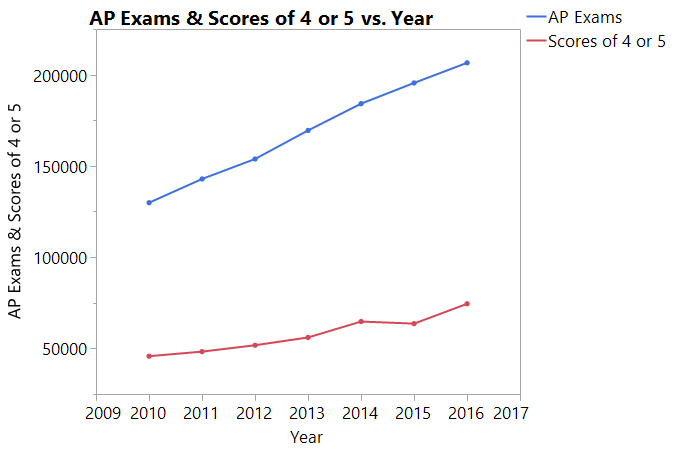


Figure 1: Growth in Advance Placement Statistics

With each new release of JMP, data management tools have grown in number and sophistication. One of these tools, **Query Builder**, is an intuitive entrée into the world of relational databases, even for users and students who have no prior background in data structures, relational algebra, or Structured Query Language (SQL).

In this paper and presentation, I walk through a representative example using sample data that ships with JMP 13 Pro, supplemented by some on-line data. The motivating question is “Can we build a useful model to understand and predict outcomes in a U.S. presidential election?”

# a plan for the investigation

Ultimately, a model to predict vote tallies will have a response (or target, or dependent) variable like “number of votes” or “percentage of votes” and might have a small number of predictor variables (factors, features, covariates, or your favorite term). With some thought, we can generate a list of possible candidate variables, and consider the types of modelling approaches that might be most suitable. Such thought processes are standard elements in an undergraduate statistics course – though the actual creation of formal multivariate models may be a stretch. Still, undergraduates can understand multivariate visualizations with, say, two fitted lines corresponding to two subgroups.

To estimate a model, we’ll need observations of the response and prospective predictors. In the Sample Data directory for JMP 13 Pro, we find two relevant data tables, which also include scripts for exploratory visualizations:

* **US Election 2008** – state by state vote counts for each candidate
* **US Demographics** – various characteristics, drawn from the Census bureau and other governmental agencies (no date provided)

In both tables, rows correspond to US states, so for this illustration we use the state as the unit of analysis. Had we wanted to analyze counties or precincts, we’d need to find other sources. Since I would like readers and viewers to be able to reproduce this demonstration, the JMP-supplied data is ideal to begin.

The demographics table has 16 variables. For this demonstration, we’ll investigate just four of them:

* Median Household Income ($)
* Population (millions)
* High School graduates (% of population)
* College degrees (% of population)

It should be noted that residents of the District of Columbia voted in the 2008 election, but do not appear in the Demographics data table. Also absent from the data tables is any indication of the relative strength of political parties in the state. As a proxy, I decided to find the party affiliation of state governors in 2008 and create a third table with that data. That process and data are described in the next section.

Once we have the desired data in three JMP tables, we’ll combine the relevant columns into a single data table using Query Builder and then begin the analysis. This demo focuses on the assembling and merging of the data.

A word on Query Builder: In JMP 12, this tool was introduced for creating database queries (see Hill, 2015). In JMP 13, it also appears in the Tables menu allowing users to build queries from JMP data tables. In this way, we can treat resident data tables as if they were part of a relational database. In my view, this is also happens to be a natural bridge to the relational model for newcomers. Before demonstrating Query Builder, let’s create the additional table of party affiliation of state governors.

# obtaining gubernatorial data

## An internet search led me to the Politdata.org website. Their Party Control Project provides data about the governorships and the composition of state legislatures as shown in Figure 2. The data are presented in a pdf, which unfortunately does not lend itself to JMP’s **File > Internet Open…** function.

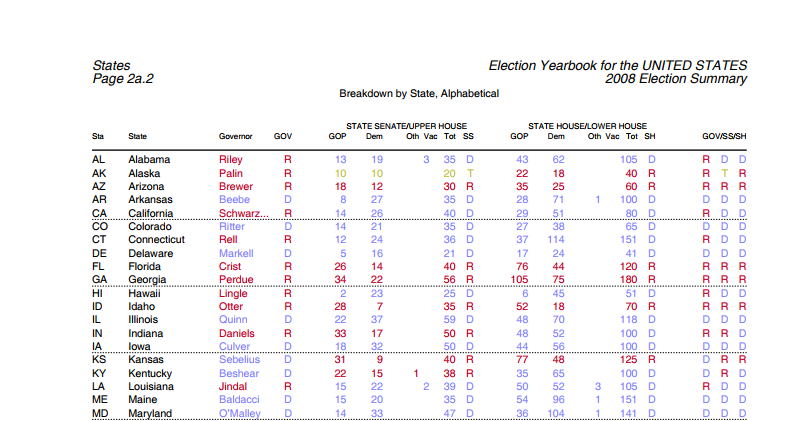


Figure 2: Polidata Party Control Project Report (partial), 2008

I copied the rectangular data area, opened a new JMP Data Table and performed a Paste with Column Names. This loads the entire block of data into column 1. The data values are separated with a single space, which would lend itself nicely to a Text to Columns operation except for the fact that there are embedded spaces in some state names and that there are empty cells in some of the pdf columns. Because I only want the first four columns of a 50-row table, it was a small matter to temporarily replace the space following *New, North, South, West*, and *Rhode* with an underscore in ten state names.

Next, I used Cols > Utilities > Text to Columns, and specified a single space as the delimiter. This split the one column into 18, and I deleted all but Columns 2 through 5. At this point, all columns share the same name, so I renamed the columns appropriately. Finally, I reversed the earlier operation and replaced underscores in state names with a single space.

# nearly ready to join

Now we have three tables, each of which contains some columns that we want to use in our modelling. Before combining them, I wanted to take a closer look at the election results data. In 2008, there were numerous candidates in the race, but Barack Obama and John McCain dominated. After finding that all other candidates typically garnered approximately 1.5% and never more than 3.3% in any state, I decided to limit the analysis to Obama, McCain, and “All Others” for this example. I modified the original US Election 2008 data table to combine the tallies for all other candidates into a single column, and created new columns giving the percentage of total vote going to the two major candidates and all others. I saved the data table as **US Election 2008 expanded**.

In class, I would note that Obama and McCain were on the ballot in every state, but some other candidates were not, and therefore there are some empty cells in the election table. This is a good opportunity to demonstrate Analyze > Screening > Explore Missing Values and to discuss issues arising from missing data. That is a topic for another presentation.

# query builder

The columns we eventually want to analyze are sitting in three JMP data tables. This is where Query Builder enters the picture. We can create a new data table consisting of just the columns we want from the three tables, and use the names of states to align the rows properly. From the teaching perspective, the visual organization of the interface is an intuitive foundation for major principles in relational database organization and operations. With tables as small as these, one (that is, a student) might be tempted copy and paste values, but that is not a strategy that scales well.

I would remind students that the data tables share the same structure: state data are in the rows and variables in the columns. Fortunately each table has a column of state names, and the rows are alphabetized by state name. These shared columns will be critical combining the data. The goal of our operation is to create a subset from the combined available data, where the resulting data table contains only the data we really want.

With the three data tables open, we select Tables > JMP Query Builder. In Figure 3, I’ve selected **US Election 2008 expanded** as the Primary table (denoted t1), largely because it contains the response variables that I wish to investigate. Eventually, I plan to estimate models for the Obama (or McCain) vote counts, vote percentages, and the binary “winner” variable (McCain/ Obama).

Notice that the selection of a table populates the lower portion of the launch window with the available columns. Later, we will select just the columns for inclusion in the query.

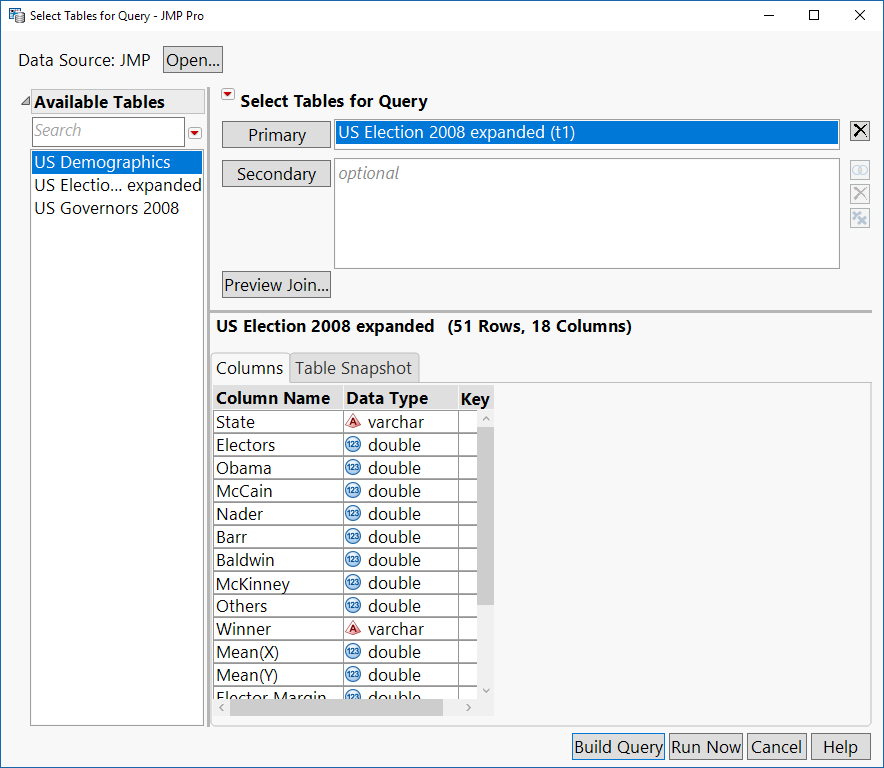


Figure 3: Initial Query Builder Dialog -- Selecting Primary Table

The lower portion of the dialog summarizes the dimensions of the data table (51 rows, 18 columns) and provides a choice between a listing of the columns and their data types or a view of the data table itself (Table Snapshot tab).

Next we select the two other tables as Secondary tables (see Figure 4), and pause to notice a few new features. First, to the right of the secondary tables we see two Venn diagram icons with the left side fully shaded. Additionally, in the Columns tab for the **US Demographics** table we now see a data element called Join and the first row of this table indicates that the Column Name **State** is associated with **US Election 2008 expanded.State** (scroll sideways to see this). JMP is proposing to join the pair of tables on the common column called State.

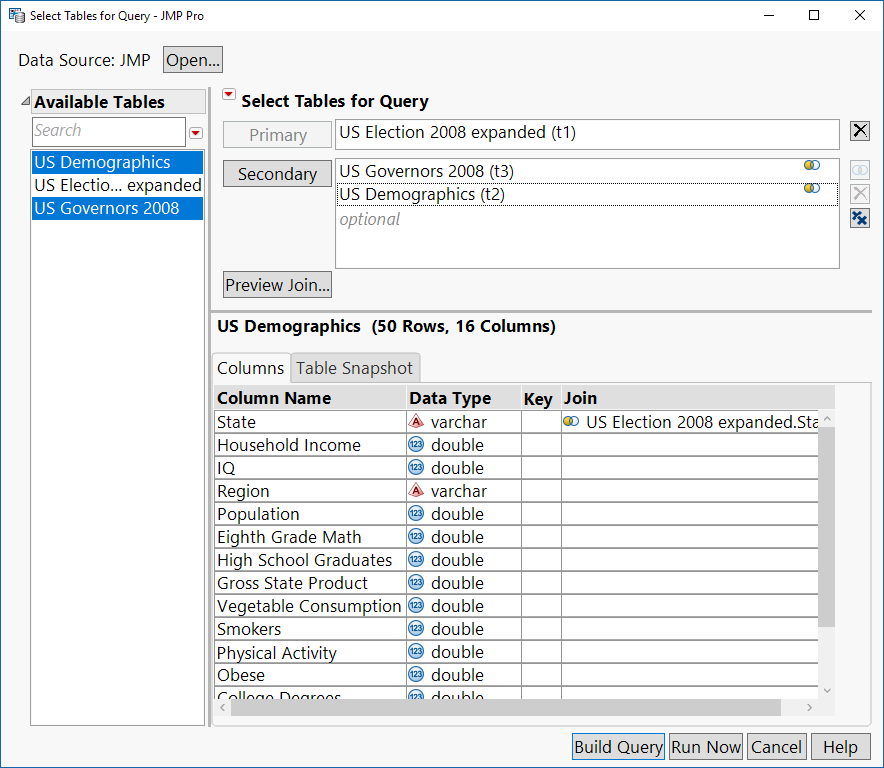


Figure 4: Selecting Three Tables

From a teaching standpoint this is quite helpful. Rather than introducing the database concept of joining two tables as a new topic that is detached from students’ other experiences, here we see it in a familiar frame of reference: it’s a way to combine columns from two tables using a column shared by both tables. It’s a small step to explain that the verb “join” has a reserved meaning in the context of databases, and that it refers to the process of matching rows according to the values of one column in each table.

I can also explain at this point that we can join using columns that have the same name (these are easy for JMP to identify) or with different names. They can be sorted differently or identically. All that matters is that there are at least some shared values in the two tables so that rows can “line up” properly without loss of information. If we want to accept JMP’s preliminary plan, we can proceed to create a query; if not, we may want to edit the planned join conditions. Although the default join is just fine here, let’s look before going to the next step.

Before moving on to select the subset of columns for our query, we pause and look at the JOIN that JMP proposes. Double-click on the Venn diagram icon to the right of **US Demographics (t2)** to open the dialog shown in Figure 5.

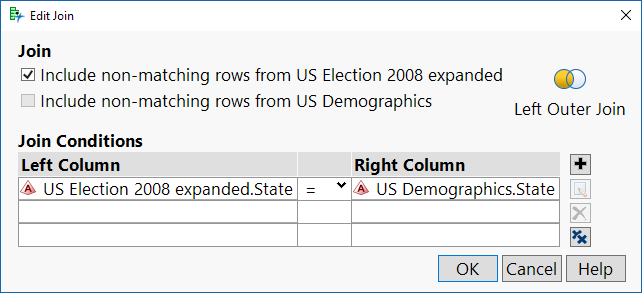


Figure 5: Specifying Join Parameters

This dialog provides a natural context to explain the difference between inner joins and outer joins. In this example, we have election results for Washington DC, but no demographic observations. We can also show that it is possible to join differently named columns, or even to have complex join conditions that rely on more than a single column.

Because the proposed Join is suitable, we can close the Edit Join dialog and click Build Query. This opens the window shown in Figure 6. The launch window layout reflects the essential elements of a basic SQL SELECT… FROM … WHERE…GROUP BY… ORDER BY… query:

* Tables in use
* Columns (variables) to select
* Aggregation computations, if any
* Conditions for filtering rows
* Criteria for ordering rows

The initial steps include naming the query (“Election Model”), specifying the source tables, and had selecting columns for inclusion in the result table.

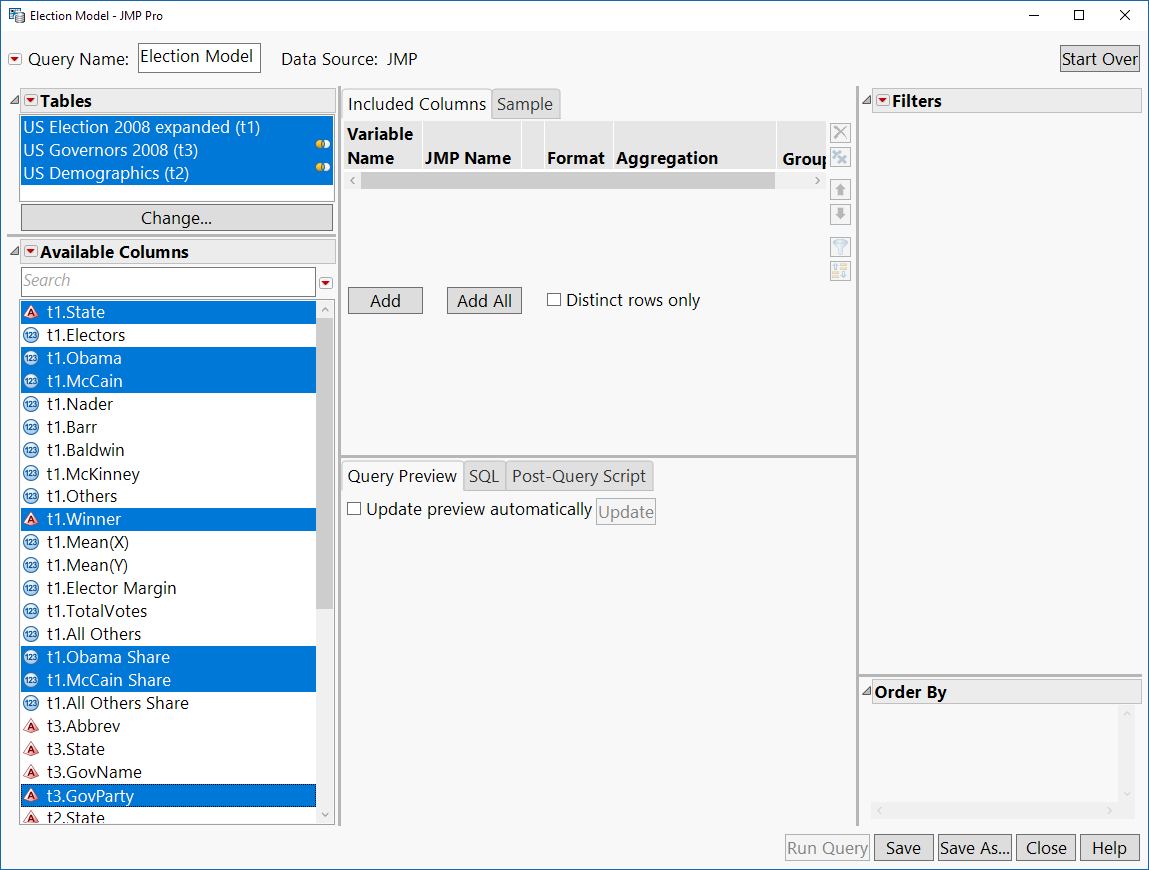


Figure 6: Starting to Build a Query

In this image, notice that the Available Columns list includes all columns from all three tables, with each column name preceded by a *tn* prefix. Ordinarily a JMP column list just shows column names, but with multiple columns, it’s necessary to also specify the source table for each column. This prefigures SQL syntax and sets the stage for the moment when we reveal the SQL code that is being written via the dialog.

Once the desired columns are chosen, click Add in the upper center section of the dialog. We see the selected columns as well as data types. There are other options as well, but for a first encounter with the subject, I defer discussion of Aggregation and Group By for a later time.

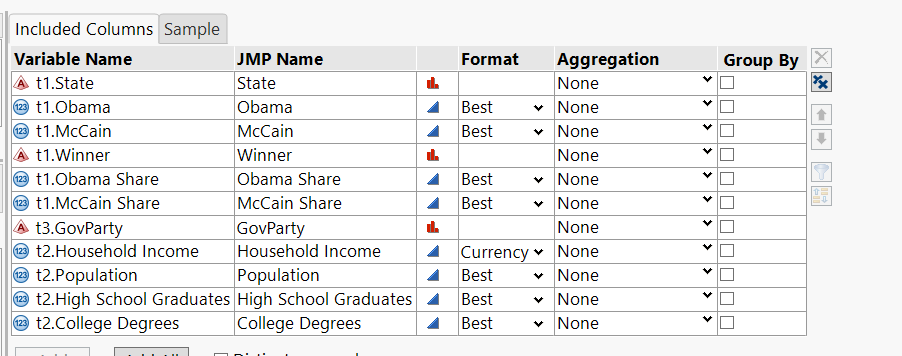


Figure 7: Summary of Selected Columns

The controls on the far right allow us to delete variables, re-order the columns, add filters (*i.e.,* subset by rows), or reorder the rows within the result table. Before going further, I also point out to students that, for the sake of reproducibility, we should Save the query.

Before running a query, there are two more things to look at in the lower center part of the launch window. First, click on **Update** next to **Update preview automatically**. This generates an image of the result table, so that we can see what it will contain.

Second, the SQL tab (Figure 8) reveals the SQL script that JMP creates based on our choices.

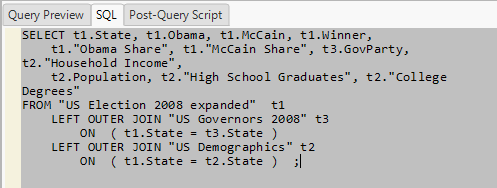


Figure 8: The SQL Code

In my own course, I would not attempt to teach SQL in a substantial way, but would point out the capitalized verbs, the recognizable table and column names, and the re-emphasize the concept of a **JOIN**. In this case, the **LEFT OUTER JOIN** is needed because there are vote tallies from 51 places and all other data just from the 50 states.

At last it is time to Run Query. The result table is also named **Election Model**, and the contents match the preview. There are several noteworthy items in the table variables pane, as shown in Figure 9, including the SQL code itself, one Graph Builder script from election data table, and controls to re-run or update the query.

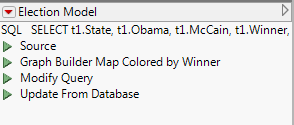


Figure 9: Table Variables Pane in the Result Table

The original election data table includes five scripts, but four of them depend on columns that did not come over in the query. Hence, only the one executable script is included now. Most importantly, the new table includes only the columns we chose for further analysis – and that analysis is left to the reader.

# Conclusions

College-level introductory statistics courses are changing. Students arrive with some prior instruction in the field, and current best practices include new topics related to data management and data preparation. Relatively few textbooks provide instruction about relational databases, and many instructors may feel reluctant to add a “non-statistics” topic to their courses. With addition of Query Builder to the JMP Tables menu, we can introduce students to this crucial subject without pain.

# REFERENCES

1. American Statistical Association (2016) *GAISE College Report* ASA Revision Committee, “Guidelines for Assessment and Instruction in Statistics Education College Report 2016,” http://www.amstat.org/education/gaise.
2. Carver, R. (2016) *Preparing Data for Analysis with JMP.*  Cary NC: SAS Press.
3. Carver, R. and Stephens, M. (2015) “It is Time to Include Data Management in Introductory Statistics”, *Proceedings of the 9th International Conference on Teaching Statistics,* Flagstaff, AZ. <https://iase-web.org/icots/9/proceedings/pdfs/ICOTS9_C134_CARVER.pdf>.
4. The College Board (2017) “Advanced Placement Program Summary Report, 2016”. <https://secure-media.collegeboard.org/digitalServices/pdf/research/2016/Program-Summary-Report-2016.pdf>.
5. Gould, R. (2010). Statistics and the modern student. *International Statistical Review, 78*(2), 297-315.
6. Hill, Eric. (2015). “Query Builder: The New JMP© 12 Tool for Getting Your SQL Data into JMP©”, available at <https://www.jmp.com/content/dam/jmp/documents/en/white-papers/query-builder-jmp-12-107669_0515.pdf>.
7. Horton, N.J., Baumer, B.S., Kaplan, D.T., Pruim, R. (2013). Precursors to the Data Explosion: Teaching How to Compute with Data. Conference presentation at Joint Statistical Meetings, Montreal, Quebec. Abstract retrieved January 2, 2014 from <http://www.amstat.org/meetings/jsm/2013/onlineprogram/AbstractDetails.cfm?abstractid=307064>.
8. JMP (2016), Version 13. SAS Institute Inc., Cary, NC, 1989–2013.
9. McCormack, D. (2015) “It’s a dirty job, but somebody has to do it… Cleaning up data in JMP®”, JMP Discovery Summit 2015, San Diego CA.
10. Polidata.org (2009) “Party Control Project: Breakdown by State, Alphabetical 2008 Election Summary.” <http://polidata.org/party/uspc08a.pdf>.
11. Steinberg, Dan (2013) “How much time needs to be spent preparing data for analysis?” Blog post, June 19. <http://info.salford-systems.com/blog/bid/299181/How-Much-Time-Needs-to-be-Spent-Preparing-Data-for-Analysis>.