# Introducing Very Large Data Sets into the Classroom — A Graphical User Interface for Teaching with Databases

**Dason Kurkiewicz, Heike Hofmann, Ulrike Genschel**
Department of Statistics, Iowa State University

## 1. Background

Over the last two decades, computing power and storage capacities have tremendously increased the amount of data that can be collected and potentially analyzed. Gigabyte size databases are not uncommon and the largest databases in the world have terabytes worth of data, e.g. search engines such as Google™ or Yahoo™, online shopping companies such as amazon™, phone companies such as at&t™ and social network sites such as YouTube™, are only a few companies that collect massive amounts of information on and around their customers. All of these are on the list of the top ten largest databases in the US (Focus Editors 2010) and companies nowadays invest significant resources to overcome the challenge of sheer data volume (e.g., the Netflix challenge won by Bell, Bennett, Koren, and Volinsky (2009)) as such amounts of data do not lend themselves easily to statistical analyses.
Having such massive amounts of data is a doubled-edged sword: computing time and computability become crucial factors in the data analysis, up to the point that they can lead to an operational breakdown of the standard analytical tools such as Excel, SAS, or R. Even when only partial information from the database would be sufficient, there is often no, or at least no easy way for sub-setting or aggregating the data at particular levels.

Furthermore, our understanding of what we consider large amounts of data has changed in recent years. The size of a data set can be defined in terms of the number of observations, the number of variables, or both. What is considered large additionally depends on the subject area (e.g., microarray analyses typically deal with thousands of observations while clinical trials usually involve a few hundred observations). In general, the transition from normal to large takes place whenever classical tools and procedures no longer work properly when performing analyses (Theus and Urbanek 2008). While we can now easily handle datasets with hundreds or thousands of records on any personal computer, massive databases are more and more common in the workplace environment and in the years ahead, statisticians and data analysts alike will more and more encounter such large data volumes at the workplace as well as the accompanying computational challenges. Thus, regardless of the actual amount of data available, the ability of effectively (and efficiently) extract information from the data at hand will remain a key skill in our profession and students who seek to become data analysts should be exposed to this real-life situation during their educational training. Consequently, it is our responsibility as educators to train students adequately and to help students become more familiar with the handling and analysis of such data sizes providing students with at

least a minimal set of basic skills and experience before entering the workforce.
A first step toward this goal is to introduce and adequately facilitate the use of large data sets in the classroom setting. Most undergraduate curricula in statistics include at least one basic statistical computing course. At Iowa State University, currently two courses are part of the required curriculum: Stat 479 "Computer Processing of Statistical Data" and Stat 480 "Statistical Computing Applications." Such courses would be suitable for introducing this topic together with the graphical use interface (GUI).

## 2. Database and Data Example

We are using the open-source relational database management system MySQL (MySQL 2010) for storage and manipulation of the data. To increase efficiency, data are often stored in normal form (Kent 1983), i.e. a large data set is broken apart into smaller tables, such that duplicative entries are avoided. While splitting the data results in more efficient storage, this imposes an additional technical hindrance due to the now non-rectangular form of the data. Further, any data retrieval from the database requires knowledge of the Structured Query Language (SQL) which, typically, is not part of the standard undergraduate or graduate degree curriculum for students in statistics or any related sciences. Additionally, many students perceive working with databases as intimidating. With the use of a graphical user interface, a gentler and more paced introduction to databases and SQL is possible due to the ease of use of a GUI over more complex (but powerful) methods. The user can access any of the GUI's functions by "point and click," similar to most statistical software introduced at the undergraduate level and does not require the acquisition of a new programming language. The knowledge of SQL is a most desired trait for those working with large data.

There are many packages out there to deal with SQL. Most require a working knowledge of SQL/MySQL to get started though and thus they are inadequate for the needs of a first time user just trying to explore databases and get a gentle introduction to how to get the desired data. One of the most well known packages is the MySQL Workbench provided by Oracle when the MySQL client is installed. For thosewith knowledge of MySQL this can provide a visually pleasing environment to send queries and save the requested data to an external file. It doesn't provide a simplified way to create a query while learning MySQL though. Another tool that provides a good interface for creating a query using SQL is SAS®Enterprise Guide®. With Enterprise Guide you do get a fairly intuitive way to subset your data and join tables. However, you must have access to SAS Enterprise Guide which is not free. Even then it doesn't show you the generated code by default and the code it does generate is for PROC SQL. It is possible to connect to external databases with Enterprise Guide but that isn't a trivial task. Within R there are the RMySQL (James and DebRoy 2010), DBI (R Special Interest Group on Databases 2009), and RODBC (Ripley and from 1999 to Oct 2002 Michael Lapsley 2012) packages (and there are probably a couple more). These are quite powerful packages that allow you to connect to databases, send SQL statements, and get the results into R. Once again, though, they aren't as explorable and instructional as would be desired for a first time user. The GUI we provide uses the RMySQL package but gives a much nicer interface for a first time user. It creates the query for the user through a point and click interface but also shows the user the created query so they can learn the language while they use it. The user is also provided a full history of the queries that were created while they are using the GUI

so they can replicate the queries made in R without having to resort to the GUI every time they want to grab the same dataset. In doing this the user learns to use MySQL.

To illustrate the use of the GUI, we chose data used in the 2009 American Statistical Association (ASA) Data Exposition (Wickham forthcoming, 2009). These data consist of flight arrival and departure details for all commercial flights within the U. S. between October 1987 and 2008, yielding more than 120 million data records and taking up 12 gigabytes of hard drive storage uncompressed, each record corresponding to an individual flight with data on 29 different variables. We gathered supplemental information from other sources, such as spatial location and runway layouts of airports, as well as operational information provided by the Federal Aviation Administration (FAA) (2010). Hourly weather information was retrieved for a subset of the major airports from the National Oceanic and Atmospheric Administration (National Climatic Data Center (NCDC) 2010) and Weather Underground (Weather Underground, Inc 2009).

## 3. Graphical User Interface

### 3.1. Description of the GUI

The GUI consists of two parts: a connection browser (see Figure 1) and the main browser (see Figures 2 to 4). Communication with the database is established through the `R` database interface (`DBI`) package (R Special Interest Group on Databases 2009) and the `RMySQL` package (James and DebRoy 2010). The `DBI` package provides the basic communication between `R` and the database, while the `RMySQL` package provides the implementation specific interface to the MySQL database (the driver).
To connect to the GUI, installation and loading of the package `dbConnectGUI` is required. This can be done by submitting the following commands in `R`

```
install.packages("dbConnectGUI")
library(dbConnectGUI)
```

Running the command

```
dbConnectGUI()
```

displays the connection browser as shown in Figure 1 allowing the user to establish a connection to the database by choosing server, user, and the database itself. While the GUI allows to connect to any MySQL database, the default information automatically connects to the ASA Expo '09 data provided by the server at Iowa State University.

Selecting 'Connect' establishes the connection to the database and opens up the main browser (Figures 2 to 4) as long as there was no error in connecting. If an error occured the user will get a popup telling the user the most likely cause for the connection error. The main browser serves the following three main functions:

1. data display for easy and immediate verification of data entries,

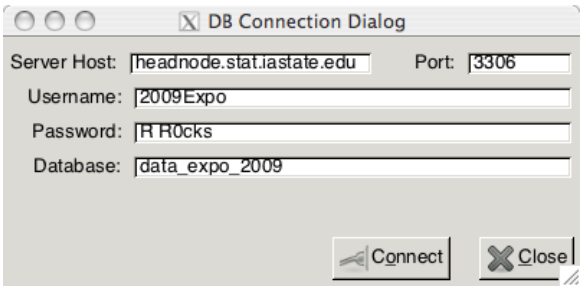



Figure 1: Dialog Window with connection information to the database. After clicking 'Connect', the connection to the database is established.

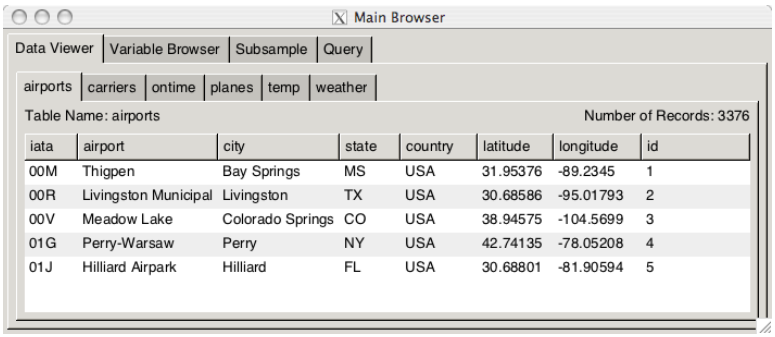
2. data retrieval/database sub-setting to prepare the data for subsequent statistical analysis,

3. learning tool for the structured query language (SQL).

The browser dialog consists of four tabs: the 'Data Viewer,' the 'Variable Browser,' the 'Subsample' dialog, and a 'Query' tab. A list of the different data tables stored in the selected database is given in form of a (scrollable) second line of tabs underneath the main four tabs. Access to any of these data tables is obtained by clicking on the corresponding data table tab; any functions of the browser dialog will now be applied to the selected data set. In our example, six data sets are stored in the database providing information on flights, airports, planes, carriers, and weather conditions.

The Data Viewer tab (Figure 2) displays available information on the first five records of the selected data set (this is similar to the command `head` in `R`) providing an overview of the data and allowing students to familiarize with the available information.

The variable browser (not shown) lists all the variables corresponding to the selected data set as well as the data type of the variables in both the SQL and R language.




| iata | airport | city | state | country | latitude | longitude | id |
|---|---|---|---|---|---|---|---|
| 00M | Thigpen | Bay Springs | MS | USA | 31.95376 | -89.2345 | 1 |
| 00R | Livingston Municipal | Livingston | TX | USA | 30.68586 | -95.01793 | 2 |
| 00V | Meadow Lake | Colorado Springs | CO | USA | 38.94575 | -104.5699 | 3 |
| 01G | Perry-Warsaw | Perry | NY | USA | 42.74135 | -78.05208 | 4 |
| 01J | Hilliard Airpark | Hilliard | FL | USA | 30.68801 | -81.90594 | 5 |

Figure 2: Data Viewer tab of the main browser dialog window. The line of tabs gives an overview of the available datasets. The first five records of the selected dataset are shown.

In the 'Subsample' tab (see Figure 3) the user can sequentially narrow the scope of the data set to a subset of interest by adding one variable criterium at a time. Criteria can be linked using either the 'AND' or 'OR' expression and any selection can be modified by removing the last specified

criterium. The number of data records in the subsample can further be limited by setting a desired sample size in the 'Limit' field. Specifying a limit will yield a selection of data records of the chosen size in the order the data records appear in the database. Alternatively, future development will extend this dialog to enable a (stratified) random selection of the records in the database.

Figure 3 shows a query from the 'ontime' data set selecting the first 100 flights (`Limit:100`) from 2008 (`Year=2008`) during the month of January (`Month=1`) that arrived in Chicago O'Hare airport (`Dest=ORD`) with at least 15 minutes of delay (`ArrDelay>=15`).

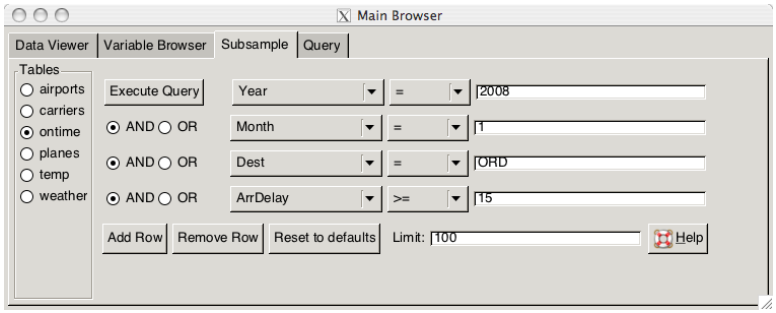



Figure 3: Subsample tab of the main browser dialog window. Subsets of the data can be chosen using a sequence of logical expressions based on variable values. The SQL query corresponding to any sequence is shown in the Query Tab (see Figure 4).

Upon choosing 'Execute Query' the above selection of observations is automatically converted into the corresponding SQL statement yielding the following SQL query:

```
Select * from ontime where  (  (  (  (       Year = 2008   )        and
                                     (      Month = 1      )  )     and
                                     (       Dest = 'ORD'  )  )     and
                                     (  ArrDelay >= 15     )  )  limit 100
```

This SQL command will be displayed in the 'Query' tab (shown in Figure 4). Logical expressions are connected sequentially using left-sided parentheses. Any later modifications to this structuring or extensions to the query in form of a more general SQL query, e.g. interconnecting several data tables, need to be made explicitly in the editable top section of the query tab (see Figure 4).

Once a query statement is specified in the 'Query' tab and having checked the 'Save in R' box, the corresponding data records can be extracted from the database and imported into the current `R` session by submitting the 'Run following query' button. The retrieved data records will also be displayed in the 'Query' tab (see Figure 4).
Prior submitting the query, the user can specify both a table name to distinguish data subsets if several different queries are run and a data file name in `R`. In the example shown in Figure 4, the data table name in the 'Query' tab and the `R` data file name is 'ORDdelay.'

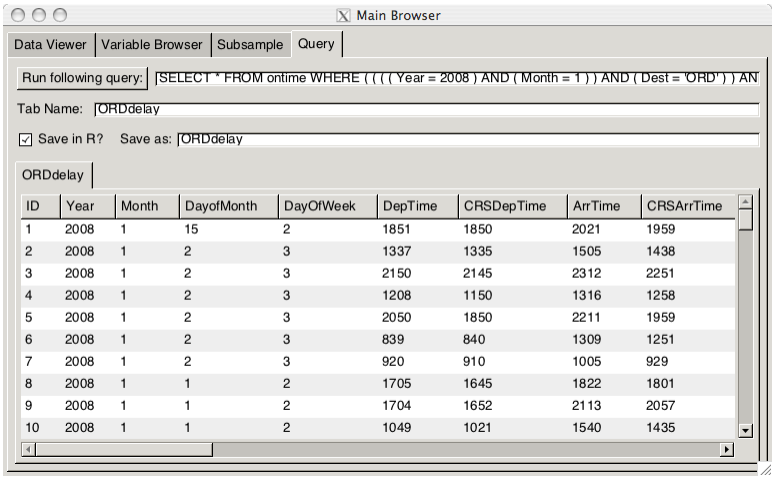



Figure 4: Query tab of the main browser dialog window. Once the SQL command is constructed (either manually or with the help of the subsample tab) the data records can be extracted from the database and imported into the R session.

### 3.2. Learning Objectives

The GUI facilitates multiple learning objectives:

1. Familiarizing students with databases and exploring the utility of databases for efficient data storage.

2. Obtaining access to data stored in databases.

3. Searching databases for specific data records of interest as well as learning how to extract such data records for further statistical analysis in R or a different software package.

4. Providing students with a first gentle and more paced introduction to the SQL language. The GUI can help facilitate the learning process and potentially improve student's attitude toward learning the database language SQL.

All or a subset of these learning objectives can be selected by the instructor depending on the background of the students and the available classroom time.
Most students have no experience with databases (see student feedback section 4) as at most universities and colleges the current training of students typically focuses on the use of statistical software packages such as SAS, JMP, Minitab or R. All of these software packages are restricted to the use of data sets of manageable size. Furthermore, instructors tend to use data sets that are typically of small size consisting only of a few variables and a limited number of data records to optimally illustrate the statistical method under consideration. Real-world data, however, is rarely available cleaned up this nicely and often tends to be much less tractable than data sets used in the classroom. For this reason, we expect that in the majority of instances the primary learning

objective will revolve around teaching students how to gain access to databases and how to subset data for further statistical analysis.

For the purpose of introducing students to the database language SQL, the 'Subsample' and 'Query' tab do provide students with the opportunity to explore the structure of SQL step by step at a pace that can be individually adjusted to the students' needs. In particular, students can receive immediate feedback upon modifying an existing SQL query: the generation of a data table confirms that students submitted a legitimate SQL query. Additionally, instructors can provide students with the final sample size of the desired data subset or subset related summary statistics to verify whether students submitted the correct SQL query.

## 4. Testing Usability of the GUI in the Classroom

In order to assess usability the GUI was tried out with students at Iowa State University, coming from two separate courses - one at the 400 level targeting Statistics undergraduates in their junior or senior year, the other one at the 500 level targeting Statistics graduate students in their freshman year. While the first course represents our main audience, students from the second course represent an 'after' audience - i.e. students who obviously have stayed in the field and have been recruited into our graduate program. A small third group consisted of advanced graduate students seeking a degree outside statistics but having an interest in data analysis knowledge as their final degree. Overall, 75 students participated in the usability survey; Table 1 shows a summary of students by status, course, and main area of study.

| Course | 400-level | | 500-level | |
|---|---|---|---|---|
| | Statistics | non-Statistics | Statistics | non-Statistics |
| Graduate | 1 | 8 | 21 | 15 |
| Undergraduate | 23 | 6 | 1 | 0 |

Table 1: Overview of students participating in the usability study by course, status, and area.

Students were initially surveyed about prior database knowledge. Results indicate that previous knowledge of database management systems differs between undergraduate and graduate students – about 20% of graduate students have had previous exposure to databases as opposed to just under 10% of undergraduate students. There is no indication of any difference in exposure to databases between students majoring in statistics in comparison to students majoring in other disciplines. Students who had prior training in database systems had an average of 1.7 years of exposure.
Because the main purpose of the survey was to assess the ease of use of the GUI as an indicator of usefulness and functionality, students were not formally introduced to the GUI prior the survey. Instead students were given a set of instructions and GUI related questions to work through and to answer while using the GUI. Although this approach may appear somewhat extreme we felt that this would yield the most informative results with respect to the user-friendliness of the GUI. The majority of questions asked in the survey is presented in Table 2. The table further shows marginal distributions of the correct answers by student status, i.e. graduate versus undergraduate students. (A complete list of all questions and the full survey is available at http://heike.wufoo.

com/forms/accessing-large-datasets/.)

Questions were chosen such that students were led step by step through the different features of the GUI but at the same time allowing the assessment of student understanding. Questions focused on three different aspects: the understanding of the GUI and its main features, the understanding of the data, and the capability to impart some first SQL knowledge. Two (graduate) students had difficulties with the interface to the extent that none of the data set and SQL related questions were answered correctly. Because these two students appeared to have struggled with the remaining course content as well, the lack of understanding, therefore, cannot be attributed exclusively to the GUI set-up. For questions regarding SQL commands (questions 9 - 11, see Table 2), 64 students were able to correctly answer question #9, 37 correctly answered question #10, and 39 students correctly answered question #11. A careful analysis of false answers submitted by students revealed that some of the erroneous answers were likely due to carelessness rather than lack of understanding. In particular, Figure 5 provides more detailed information on the type of mistakes made by students and summarizes errors by error sources, e.g. for question #10 wrong use of logical expressions in the 'Subsample' tab, failure to use the correct variable name, and mis-specified text values were leading causes of errors (the category 'value' indicates mis-specified variable names or values while the category 'logic' indicates wrong use of logical expressions).

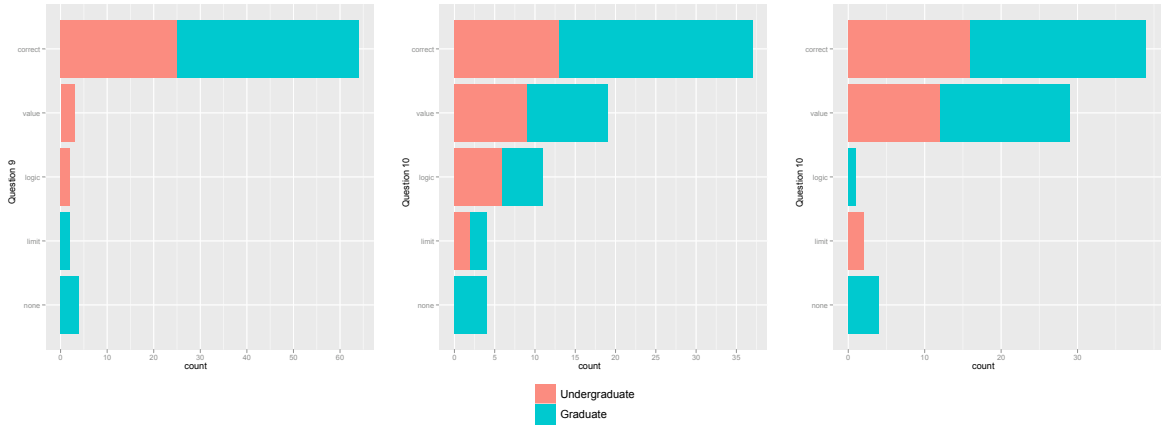



Figure 5: Students' feedback on questions 9, 10, and 11. Wrong answers are classified according to source of error. 'value' and 'logic' can potentially be considered carelessness errors rather than mistakes due to lack of understanding of the GUI. Categories 'limit' (specifying a wrong limit of data records) and 'none' (not providing any answer) are more serious errors and indicate a lack of understanding of the GUI.

Figure 6 shows results regarding the perceived importance of the task and how useful students felt the GUI was in completing the task. Although graduate students show a more clear understanding of the importance of database knowledge as compared to the undergraduate students, both groups respond similarly about the GUI's usefulness to complete the task.

Figures 7 and 8 illustrate that the GUI successfully facilitated the completion of the task. The task difficulty was clearly perceived higher by students before the use of the GUI than afterwards. Figure 7 shows marginal distribution of assessed class difficulty, while the fluctuation diagram of

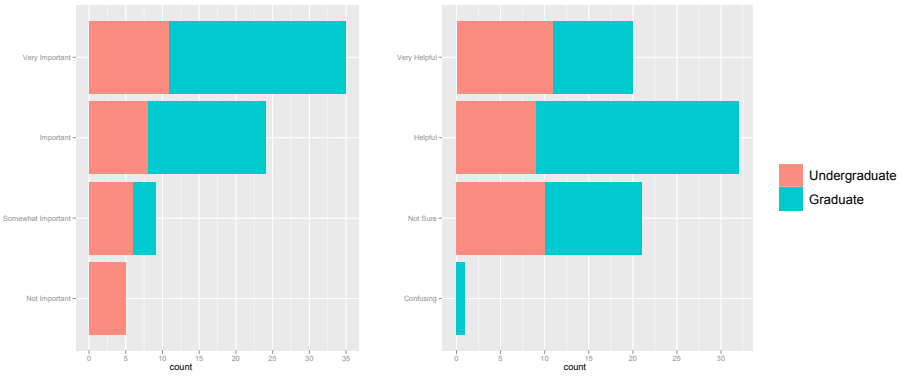



Figure 6: User feedback: perceived importance of task (left) and usefulness of the GUI to help with the task (right).

figure 8 shows the joint distribution. Very few students find the task harder than anticipated - we take this as an indicator that the GUI is a useful resource for introducing students to databases as intended.

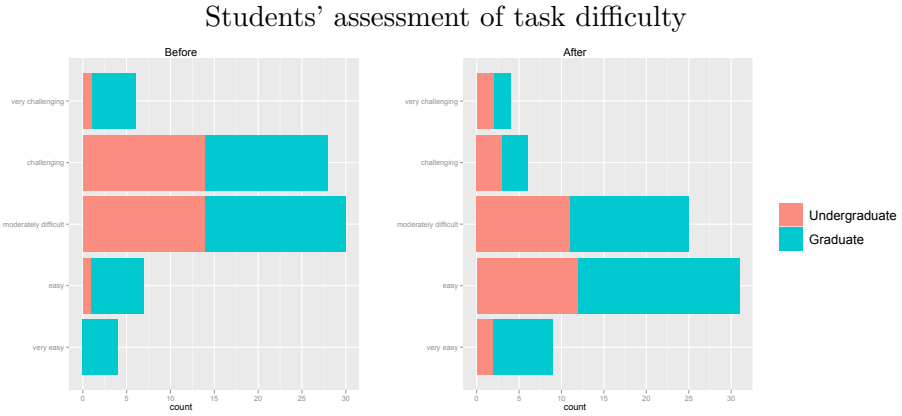



Figure 7: User feedback: anticipated degree of difficulty is greater than experienced difficulty of the task.

## 5. Dissemination and Developmental Outlook

To achieve a broad dissemination of the developed GUI, the database is accessible for public use and stored on a server of the Department of Statistics at Iowa State University. An accompanying webpage is available at http://www.public.iastate.edu/~hofmann/vldb.html. This webpage provides the following information and tools:

1. a link to the `R` source code for the GUI - also available in form of the `R` package `dbConnectGUI` available on CRAN,

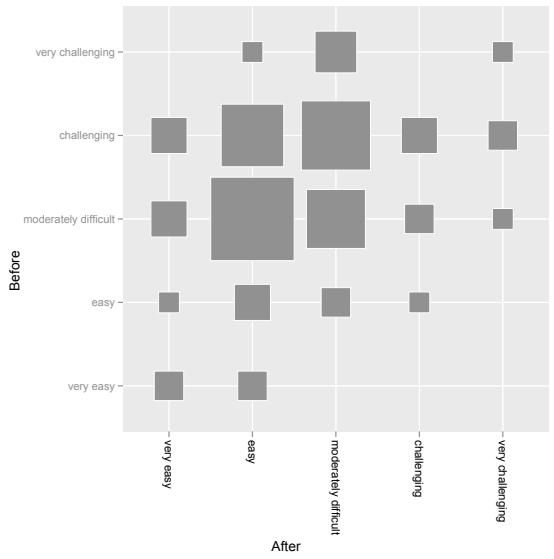



Figure 8: Fluctuation diagram of students' assessment of task difficulty before and after completion of the study. Generally, students find the task easier than they anticipate.

2. information on accessing the SQL database for the '09 ASA Data Expo,
3. a detailed description of the data,
4. examples and sample analyses for classroom demonstration,
5. a growing list of links to other databases accessible on the server at Iowa State University,
6. contact information for feedback.

In particular, we seek to implement more of the SQL functionality into the graphical user interface. While the query tab allows the full spectrum of the Select Query Language, we have only implemented the interface to data sub-setting. We are preparing to also support a user interface for joining data across several tables, and allowing stratified sampling. Other databases that are currently available from the server at Iowa State University include information on how and where money made available through the American Recovery and Reinvestment Act 2009 (ARRA) is being spent. This is data made publicly available by `recovery.gov`, which was being used in Sunlight Labs (2010)'s 'Design for America' Challenge. More information about the data is available through Graphics Working Group – Iowa State Statistics (2010). Once the GUI also supports random sampling, it can also be used as a data resource for instructors allowing, for example, to sample data sets individually for classes or students that will yield different numerical results in a statistical analysis without changing the context of the data or task, while at the same time allowing for an automatic assessment of correctness of answers.

We introduced the idea of a graphical user interface that allows an introduction of data sets into classroom that otherwise are too large in size to be managed by commonly used statistical software.

| Question | Correct Answers (%) | |
|---|---|---|
| | U'grad. | Grad. |
| **Questions regarding the Interface** | | |
| 1. Upon opening the GUI, you should see two lines of tabs. What is the default tab opened in row 1, what is the default in row 2? | 100.0 | 95.6 |
| 2. List the first two airports and cities (case records) that are displayed. | 100.0 | 95.6 |
| 3. How many rows of case records are displayed by default when opening the GUI? | 96.7 | 84.4 |
| **Questions assessing basic understanding** | | |
| 4. Does the dataset contain more case records than the ones displayed? | 93.3 | 93.3 |
| 5. The ontime tab contains all the flight information. What is the total number of flights in the database? | 80.0 | 91.1 |
| 6. Which data table has the fewest variables? | 100.0 | 95.6 |
| 7. Which data table has the most variables (don't count, just eyeball)? | 93.3 | 95.6 |
| **Advanced questions: questions assessing SQL understanding** | | |
| 9. Select the 'ontime' radio button. This data set contains information on all commercial flights over the last 20+ years. Specify to select a limit of 100 flights in 2001. Once you're done, press 'Execute Query'. The GUI will change to a view (the Query tab), where you can see the resulting query. Copy the query and paste it into the space below: | 83.3 | 86.7 |
| 10. Change back to the 'Subsample' tab. Change your sample. Remove the limit, but now only select flights into Des Moines (DSM), that were delayed by at least 60 minutes on arrival - the year is still 2001. Again, execute the query and paste the resulting SQL statement into the space below: | 43.3 | 53.3 |
| 11. Without the help of the GUI change the SQL statement from question 10 such that you only regard flights that were not delayed upon departure (less than 15 min delayed). Paste the resulting query into the space below: | 53.3 | 51.1 |

Table 2: Questions of the usability study, marginal distribution of correctly answered questions.

We explained some of the technical and pedagogical aspects of the GUI and an extension of the GUI is currently under development. Although the GUI has been used successfully in the classroom a full implementation and evaluation of the GUI is still necessary and currently being prepared.

## 6. Future Work

The GUI as it is provides basic subsetting functionality which is a good start to introducing students to the SQL language. The usability study provides evidence that the GUI was helpful in facilitating a new user's ability to learn and SQL and access databases (at least in a minimal way that functionality was provided for). It is planned to add the ability to grab random samples from the database with the possibility of stratifying or selecting the random sample based on some subsampling criterion. This will allow instructors to easily grab a different dataset for each student to create customized assignments or projects. If an instructor combined the ability to grab nice

random stratified subsets of a large database with a tool like Sweave or knitr Xie (2012) the creation of customized assignments would be an accomplishable task.

One of the main reasons to put databases in normal form Kent (1983) is to save space and provide the ability to grab only the relevent information. However, this breaks up the data so that not all the interesting information about an observation is in one row of a database. To get the data back into this form it would be required to perform a 'join' on different tables in the database. This functionality is not yet part of the GUI and would be a most welcome addition. This is being worked on and would require a more advanced layout and it is not trivial to make it easy to perform a join for a new user. For the sake of introducing databases and SQL it was decided that not having this functionality was fine, however, for a more advanced user this is a desired feature.

## References


Bell R, Bennett J, Koren Y, Volinsky C (2009). "The Million Dollar Programming Prize." *IEEE Spectrum*, pp. 28–33.

Federal Aviation Administration (FAA) (2010). *Aviation Data and Statistics*. URL http://www.faa.gov/data_research/aviation_data_statistics/.

Focus Editors (2010). *10 Largest Databases in the World*. URL http://www.focus.com/fyi/operations/10-largest-databases-in-the-world/.

Graphics Working Group – Iowa State Statistics (2010). *Making a Full Recovery*. URL http://www.stat.iastate.edu/DesignForAmerica/Recovery/.

James DA, DebRoy S (2010). *RMySQL: R interface to the MySQL database*. R package version 0.7-5, URL http://CRAN.R-project.org/package=RMySQL.

Kent W (1983). "A simple guide to five normal forms in relational database theory." *Communications of the ACM*, **26**(2), 120–125.

MySQL (2010). *MySQL tools*. URL http://www.mysql.com.

National Climatic Data Center (NCDC) (2010). *Digital Data Files*. URL http://www.ncdc.noaa.gov/oa/ncdc.html.

R Special Interest Group on Databases (2009). *DBI: R Database Interface*. R package version 0.2-5, URL http://CRAN.R-project.org/package=DBI.

Ripley B, from 1999 to Oct 2002 Michael Lapsley (2012). *RODBC: ODBC Database Access*. R package version 1.3-4, URL http://CRAN.R-project.org/package=RODBC.

Sunlight Labs (2010). *Design for America*. URL http://sunlightlabs.com/contests/designforamerica/.

Theus M, Urbanek S (2008). *Interactive graphics for data analysis: Principles and examples*. Chapman & Hall, London.

Weather Underground, Inc (2009). *Historical Weather Records*. URL http://www.wunderground.com/.

Wickham H (2009). *ASA Data Exposition 2009*. URL http://stat-computing.org/dataexpo/.

Wickham H (forthcoming). "ASA Data Exposition 2009." *Journal of Computational and Graphical Statistics*.

Xie Y (2012). *knitr: A general-purpose package for dynamic report generation in R*. R package version 0.2, URL http://CRAN.R-project.org/package=knitr.