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,

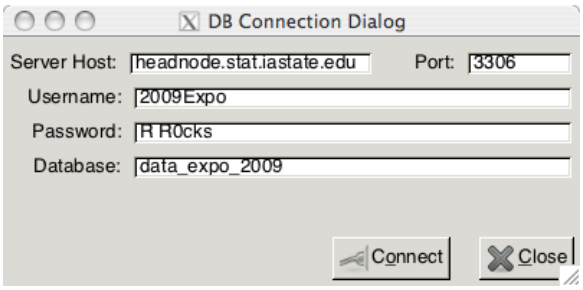


Figure 1: Dialog Window with connection information to the database. After clicking 'Connect', the connection to the database is established.

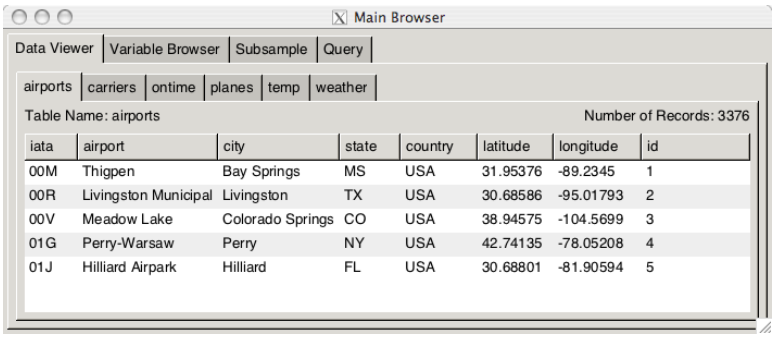
2. data retrieval/database sub-setting to prepare the data for subsequent statistical analysis,

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`).

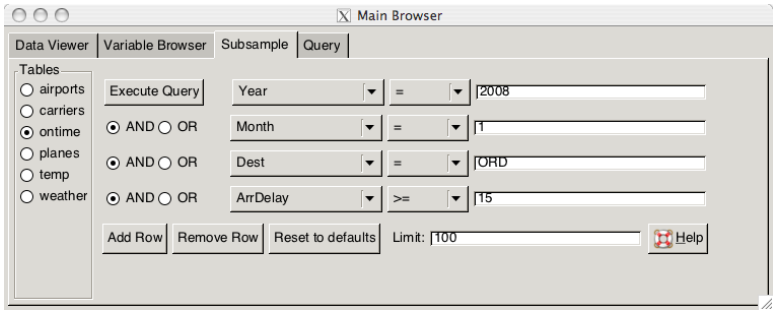


Figure 3: Subsample tab of the main browser dialog window. Subsets of the data can be chosen using a sequence of logical expressions based on variable values. The SQL query corresponding to any sequence is shown in the Query Tab (see Figure 4).

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.'

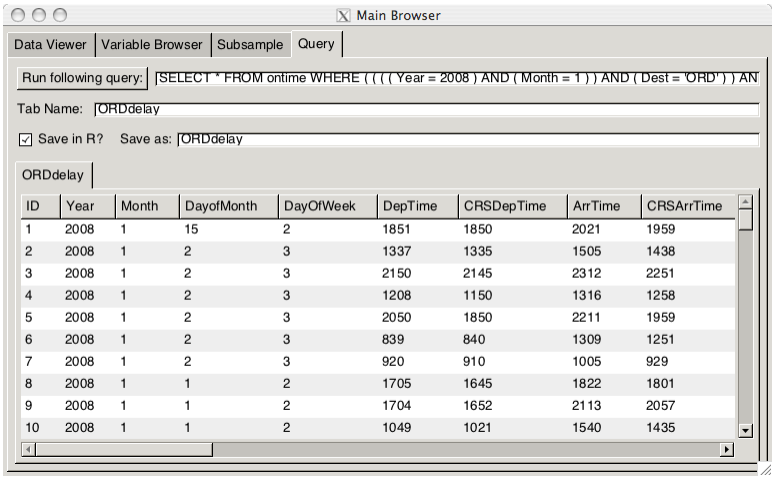


Figure 4: Query tab of the main browser dialog window. Once the SQL command is constructed (either manually or with the help of the subsample tab) the data records can be extracted from the database and imported into the R session.

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).

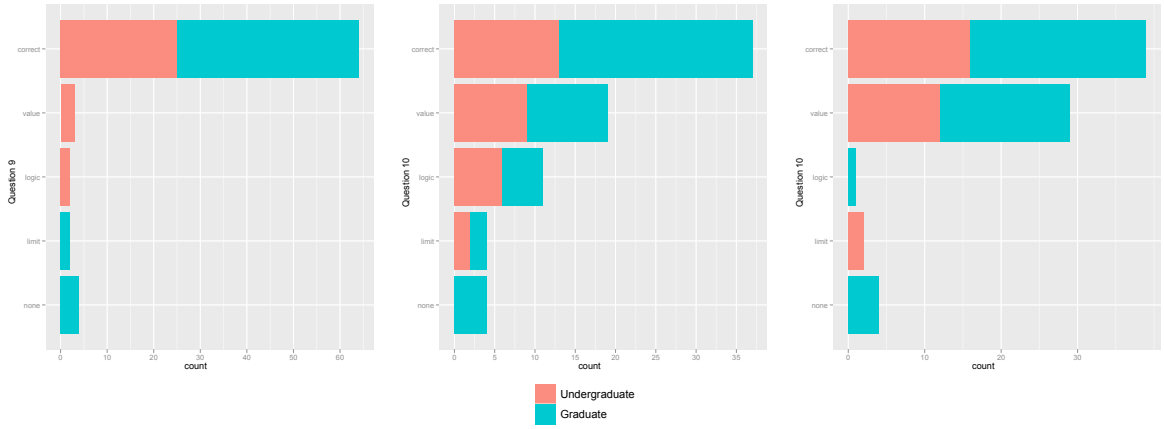


Figure 5: Students' feedback on questions 9, 10, and 11. Wrong answers are classified according to source of error. 'value' and 'logic' can potentially be considered carelessness errors rather than mistakes due to lack of understanding of the GUI. Categories 'limit' (specifying a wrong limit of data 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

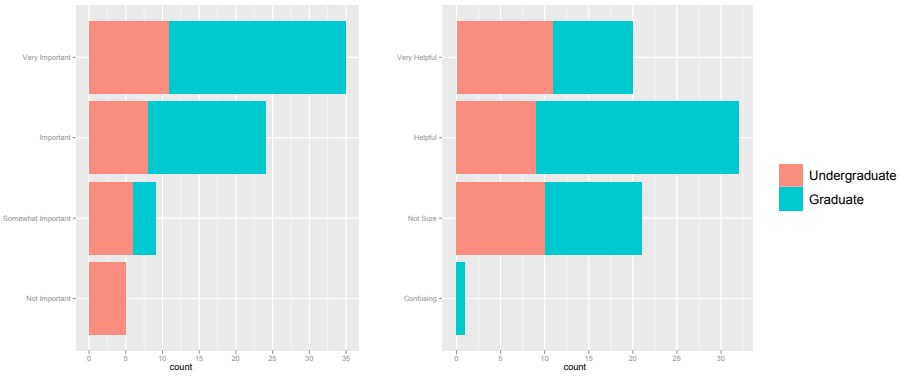


Figure 6: User feedback: perceived importance of task (left) and usefulness of the GUI to help with the task (right).

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.

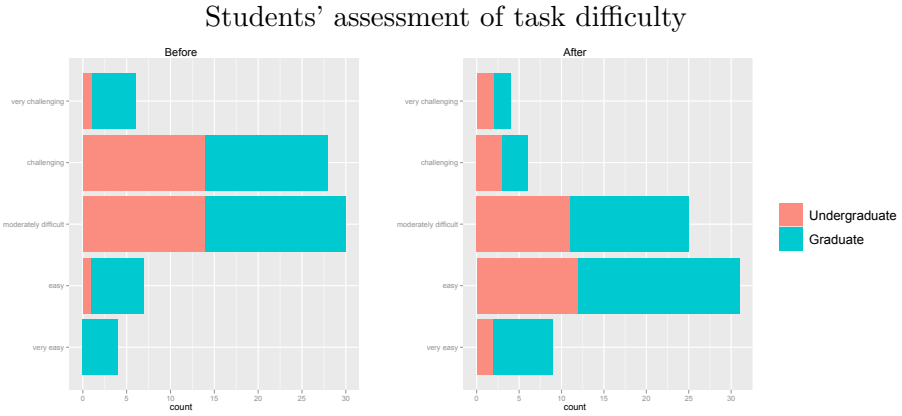


Figure 7: User feedback: anticipated degree of difficulty is greater than experienced difficulty of the task.

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,

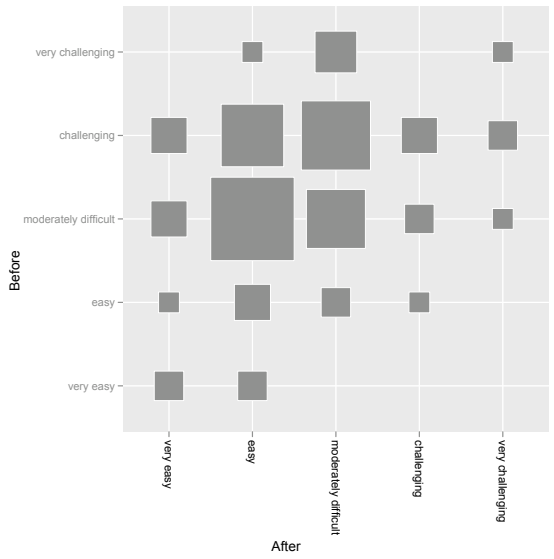


Figure 8: Fluctuation diagram of students' assessment of task difficulty before and after completion of the study. Generally, students find the task easier than they anticipate.

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.

7. Appendix

7.1. Review1

View Review

Review by: Reviewer #1

Review date: Apr 29 2011 09:20 AM PDT

The review:

Basically, the paper discusses an effort to expose juniors/seniors and graduate students to large data in statistical computing courses. The authors use a database to hold large datasets and provide the students with access to the database via a GUI for R that they wrote. The GUI is the primary focus of the paper. The paper goes on to analyze and interpret evaluations of the GUI from the students.
The work is fine and may be useful to some instructors. However, I don't think it is very compelling from a pedagogical perspective or a general description of, or introduction to, technology. As a result, I don't think it is appropriate for TISE and recommend rejecting it. I would suggest that the paper be shortened (by omitting the student evaluation) and submitted to the R Journal. However I think the GUI is too simplistic to be of much use and not necessarily appropriate for the R Journal. I am happy to re-review this if the authors can make a compelling intellectual argument for the pedagogical benefits of the approach and make the GUI significantly more general.
The premises of the paper seems to be

a) it is important to see large data,
b) a GUI is necessary to make this easy. I agree with the first premise and applaud the authors for introducing this topic into their stat. computing course(s). I don't buy the second premise. A GUI can be useful in introductory classes where we want to hide all the computational details. But this paper is not addressing intro. stats classes but students more advanced in their studies. (The paper does take a very long time to clearly identify the type of students in question.) The students are people taking stat. computing courses. There is a merging of two concepts - a) large data, and b) accessing it through databases. This is one of the places where the paper falls down I believe. It is using the database system merely as a way to access large data but not leveraging or supporting any of the real aspects of a database, e.g. joins. So the paper is merely indicating to students that one can access large data from,

b.a., R. That's a useful lesson, but it can be taught in many different ways

which are more interesting and valuable pedagogically. Furthermore, one can work with large datasets in SAS and use lots of interesting statistical methodology on that large data. Similarly, one could hide all of the database concepts and details. Several ways come to mind that lead to the student find out about the existence of richer ways of dealing with large data and/or databases. a) using bigmemory or external files or sqldf or RSQLite, b) using classes in R to represent SQL tables and then having methods for basic statistical operations such as mean, sd, cor, etc. c) using overloaded operators in R so that an expression such as dbTable[(var1 < 10 & var2 > 20) | var3 %in% c("Jan", "Feb", "Mar"),] would be transformed under the covers into an SQL query and executed. dbTable would be the table in the database. This is a richer approach as it supports the use of variables in different tables and hence joins. Furthermore, it builds on the familiar R syntax and concepts that the students are expected to know. I vaguely recall that there was a functioning prototype done for c) about 10 years ago as part of the Omegahat project. By using the simplistic GUI to a database, there is a significant chance that students will misunderstand the concepts and power of databases and think of them in this very limited manner. I don't think the students need or are well served by a GUI, specifically this GUI.

c) This GUI is way too simplistic, currently only supporting simple logical conditions, e.g. WHERE x < 10 AND y > 20 OR y < 10. It cannot manage SQL queries that involve variables in two tables and so cannot do basic joins
   - the bread-and-butter of databases. While simple logical conditions are powerful for working with single tables that are large, there are many other commonly used features of SQL that one would need to understand in "a real-life situation", as the paper lays claim to. As result, the GUI is for very simple queries of a single table, the equivalent of a very large data frame in R and the GUI is not, in my opinion, the best way to achieve "transparent" access to such data. Adding support for variables in different tables is not feasible with the current UI design and is very challenging. So this GUI won't be able to "get there from here".

ii) The GUI is fine if there are no errors, but if there are, the student may well be more confused by the presence of the GUI and have to resort to using SQL. Even logging in can result in an error and informative error messages and guidance must be provided. This is a common flaw in many ad hoc GUIs.

iii) Most importantly, the GUI attempts to hide the intellectual concepts of a database and working with large data. These are the lessons that the students need to learn for the experience to be useful. These include the efficient organization of a data in multiple tables indexed by a common variable, different types of joins, working in blocks on the result set. These are the general and intellectual concepts.

The authors talk about the need for students "to be exposes to this real-life situation". There are other GUIs, mostly commercial and professional, for interacting with databases and constructing queries via visual programming languages, i.e. GUIs. The GUI described here is quite limited and intended for pedagogical use. As a result, the students are very, very unlikely to use this GUI in "real-life". Section 2 suggests that the GUI is appropriate because it is "similar to most statistical software introduced at the undergraduate level". Since the papers suggest using this in a stat. computing course, surely the students will be using a programming language and not just point-and-click GUIs. Secondly, this statement mixes all students together from intro. stat to majors. More focus on the type of students is needed. Furthermore, on page 3, the paper describes how to install and run the GUI using R commands. So the students do need to be able to use the R command-line. All of this indicates that there is some confusion in the pedagogical thinking and framework of the authors.

Several of us do teach SQL in our undergraduate stat. computing courses (upper-division) along with other Domain Specific Languages (DSL) such as regular expressions, XML and XPath, and even regular programming languages such as Python or Perl (in graduate classes mostly). I have a strong belief (and experience) that the students gain more from doing things via a command-line language, ideally in R, that allows them to explore these DSLs in a manner in which they can reason about the language and its operations. The fact that students can provide "extensions to the query in form of a more general SQL query" by editing the generated SQL code in the Query tab is silly. If you haven't taught them to reason about SQL, how can they modify the SQL query in any reasonable manner with any knowledge of what they are expressing. The paper claims that the GUI facilitates "familiarizing the students with databases". I don't think this is true. They may become familiar with accessing a single large rectangular table which happens to be in a database. This could just as well be done via a password protected spreadsheet or NetCDF file and appear the same to the student.

Statistical Computing has intellectual content and merit in our programs. We need to teach this rather than a bunch of how-to's or tricks. George Cobb put this well when he said it reminded him of the way Math depts. thought about statistics 35 years ago as just a set of how-to's based on real math. So I think this GUI and the claim to familiarize students with databases does a great disservice to the students. I strongly recommend trying to teach them the basics of SQL instead. If it is just exposure to large data, then evaluate different ways to do that without tieing it to databases and confusing implementation with pedagogy.

One could make an argument that the GUI is a useful approach to dealing with non-majors taking upper-division stat. courses. I am not convinced it is a very compelling argument, but it may be more appropriate than for thos taking stat. computing courses. The work is fine and I hope the authors learned a great deal about GUIs and databases. It is great to have people exploring these more.

However, this seems more at the level of an undergraduate or master's project for a class rather than academic-level research. Not all activity related to teaching is worthy of publication or even the best way to teach. The paper is not very clearly written and lacks clarity and precision in discussing R and database concepts. It talks about using "the GUI" without first describing/mentioning it in the text, only the title.
Specifics:
Why doesn't the dbConnector package initialize the GUI and remove the need for the student to call the DatabaseConnect function, since no parameters are needed? So does it live up to its pedagogical purposes? The type of students is alluded to at the end of the introduction, but not clarified until page 6. As a result, readers will still be unsure whether we are talking about upper-division or lower-division students. The authors talk about 400 and 500 level classes. These only make sense within their own university and suggest a certain insularity. The terms upper-division and lower-division would be much more meaningful. The tick labels on the graphical displays/plots in the paper are unreadable. References in the introduction related to AT&T and people from the Augsburg graphics group. The references are a little oblique and indirect and are "collaborators" or friends of the authors. The dataset (and a good one) was collated by a former student of the authors' department. It would be much better to have more widespread and appropriate references. References to the DBI package should be to the package and certainly not a mailing list!
Section 2: One has to introduce RDBMS to the readers and define "normal form" more completely and correctly. This is a casual description that would confuse people who aren't already familiar with the concepts. The description of the browser dialog that mentions a data set rather than a table suggests confusion or very imprecise phrasing. Indeed, the concept of joining across tables (their "data sets") is missing in the entire paper. The description of the DBI package (page 2) is a little casual and incorrect. The package provides the generic architecture that is implemented by the database-specific packages such as RMySQL, ROracle, RSQLite, etc.
The connection browser tries to make logging in easy for the student. But the student can easily forget the name of the database to be used or mis-type the server name or the user name or password. At the very least, after specifying the server, login and password, a menu of databases should be added by querying the server for the databases for which the user has privileges. I would imagine that if this were to be used by naieve students, one would want to customize this so that the defaults were set to the specifics for the course and as a result, one could by-pass this step altogether. So either the students need more help or they are able to master following command line instructions. There is no discussion of error handling during the connection. The password should not be displayed as clear text. Use a password text entry widget. The GUI should display the schema for a table, i.e. the table listing the variables and their types. This would improve the pedagogical nature of the GUI, making the student aware of the strongly typed nature of the DBMS With the "Save to R"

```
check box, is the result set really save to a "data file name in R" or
should this be correctly stated as a variable name? Personally, I find the
expectation of the student using the GUI to access the data and then going back
to the R prompt and work with the R command-line to use that data a little
contradictory. Either they know how to use R and don't need the GUI (and use
some R language mechanisms to access the database transparently) or they need
the GUI and so cannot be expected to use R in which case the GUI should do a
great deal more.
The student evaluations are a nice addition, but of course there is the
possibility of bias as the GUIs authors are the instructors. Furthermore, I am
not sure the questions asked of the students do tell us about the effectiveness
of the mode of pedagogy, i.e. the GUI.
```

7.2. Response to Review 1

The reviewer mentions that they would be happy to re-review if the authors could make a compelling argument for the pedagogical benefits of the approach and make the GUI significantly more general. We agree that the GUI could be more general and the future work section shows there is intention to expand the GUI itself. However, we still believe there is value to the GUI for those first learning MySQL. Some people feel that a GUI feels much more natural. If you throw them in the wild with a bunch of code they'll just run and hide. Clearly for these types of people this isn't the approach we want to take. Pedagogically the GUI helps the user learn the language in a more comfortable environment while simulateously allowing the user to grab queries from a database. Adding JOIN statements would certainly enhance the GUI and with databases prefering to be in a normalized form this would certainly be advantageous. As mentioned previously this presents a technical challenge to lay out the GUI in a way that is both easy to use and not too overwhelming for a first time user.

The reviewer mentions "I don't think the students need or are well served by a GUI, specifically this GUI". It was never stated that all students like or prefer GUIs. For some it is beneficial to see the results of a query in the form of a GUI. The code is generated for them but they see the generated code and get a copy of the code. I believe that there is pedagogical value to this GUI and the usability study suggests that there is benefit to using the GUI to introduce MySQL.

The reviewer mentions "There are other GUIs, mostly commercial and professional, for interacting with databases and constructing queries via visual programming languages. The GUI described here is quite limited and intended for pedagogical use. As a result, the students are very, very unlikely to use this GUI in 'real-life'." We are fine with the reviewer having this opinion. The GUI was created for pedagogical purposes and there isn't much intent to completely replace any commercial tool. However, as described at the beginning of the paper this GUI does provide a fundamental set of properties that no other package known to the author provides. The main goal is to introduce the user to SQL in an environment that is friendly to them while providing a fundamental set of tools that makes it easier to understand the database structure. Currently the GUI is limited to the essential operations that we would like the user to be able to do. If a new

user is scared to even try MySQL they aren't well suited by something that gives them a visually pleasing text box to submit queries.

The reviewer mentions "Furthermore, on page 3, the paper describes how to install and run the GUI using R commands. So the students do need to be able to run the R command-line. All of this indicates that there is some confusion in the pedagogical thinking and framework of the authors". One reason this path is chosen is due to necessity. It's a fairly simple process to get it installed and we do believe the user can copy and paste some code if they need to. We definitely see it as a possible for a user to know R but still be a little wary of learning MySQL and would prefer a GUI. There are a lot of students that get introduced to R in a stats course and only learn what they need to know - they aren't likely to touch SQL and might not be comfortable with the coding in the first place. We aren't advocating the GUI is used by everybody but we don't want to alieniate users that want to grab data off of databases but aren't comfortable with MySQL yet.

The reviewer mentions that they SQL in their undergraduate stats computing courses. We think this is great. They might not as much use for the GUI we provide. Not all institutions are graced with students that have those levels of computing ability or don't have a curriculum that allow for such flexibility. A GUI that is easy for a user to explore could be a reasonable alternative for such institutions.

It does seem like the reviewer has a bias against GUIs in general. To be honest not all the authors are fans of using GUIs either. However, our GUI facilitates the learning process by intuitively allowing the user to deal with databases and see the commands that are used to generate them. This gives some exposure to the code. A user can see an example of what the database looks like through the 'Line Viewer' and can explore the fact that there are different tables that can be accessed. Just because a reviewer dislikes GUIs in general isn't a reason to shoot one down that has pedagogical value and allows users that aren't as comfortable with code to access databases.

The reviewer claims "Either they know how to use R and don't need the GUI (and use some R language mechanisms to access the database transparently) or they need the GUI and so cannot be expected to use R in which case the GUI should do a great deal more." Once again the reviewer fails to consider a portion of the target audience - that that might be comfortable with R but don't know SQL and prefer GUIs. There is also the group that barely knows R but still needs to access a database. Should the user be expected to become perfectly proficient with both R and SQL in a short time frame? Why not use a tool to ease the learning of one of the languages (SQL) and make the experience more enjoyable for the new user? We believe some users aren't as comfortable with the command line as the reviewer is and would appreciate the guidance that a GUI gives. The usuability study implies this much.

The reviewer does make some good points and some helpful suggestions and we appreciate that. However, they were harsher than needed to be and they gloss over what we feel is the main part of the paper. The student evaluations suggest there is value to a GUI. The pedagogical nature of the GUI was avoided and the technical aspects were focuses on.

7.3. Review2

Review of
Introducing Very Large Data Sets into the Classroom:
A Graphical User Interface for Teaching with Databases

The paper raises an important issue – that while interesting and increasingly available, the complexity and sometimes sheer size of large data sets makes them difficult to deal with both technically (because of potential computational breakdowns) and conceptually (because of the number of variables, often mashed or linked in smaller tables). Unfortunately, rather than pursuing the implications of how to deal with such data sets broadly, the paper focuses on the very narrow solution of extracting samples from large data sets to address computational (but not conceptual) complexity, and then explores a specific graphical user interface for doing so. In quickly moving to the GUI "solution", the paper loses sight of the much more interesting larger issues, and presents itself more as a research report on a small technical innovation rather than as a position paper. As such, many of the guiding questions suggested for review of a position paper are not relevant.

After correspondence with the TISE editor, we've decided to review the paper as a Technology Innovation rather than as a Position Paper. Though this doesn't resolve all of this reader's concerns with the paper, however, as noted below, it seems a better fit with the goals and approach of the paper.

The GUI innovation described in the paper is intended to solve the problem of providing technical access to complex large data sets, though it doesn't address how students can be using the tool to think about the larger data set and what it's saying. The authors state (p.1, ¶2) that "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." This seems to be the core of the problem they are addressing, but it is only partially addressed with their GUI.

The authors describe the components of the R packages used to access the off-site database, and provide specific steps for doing so. They walk the reader through use of the dialogs and tabs that make up the GUI. They don't describe how to set up the GUI for use with other data sets (though presumably it's designed for flexible use), and especially how to make links among conceptually related databases that were not collected together. Such descriptions would increase the utility of the paper.

Though the authors describe their innovation sufficiently well, it is not clear how this example of a query tool is particularly innovative. It allows for searches of subsets of variables linked by Boolean "AND" and "OR" operators – it's not clear if there is also a "NOT" operator to exclude specific classes or cases. It allows users to limit the size of the result set (but it's not clear if this is the *first* N items matching the query, or a random subset, or something else.) It then translates the search options entered using drop-down menus and numerical input into SQL syntax. On one hand, it seems the intention is to use the more intuitive interface to help students learn to write SQL syntax directly, though the display of the syntax is limited to a single line (presumably scrollable) which, however, de-emphasizes the importance of this aspect of the tool. That is, if the purpose

of the GUI is to learn SQL, it's not clear that it's designed very well as a teaching and learning tool.

The authors state four Learning Objectives for the tool. It clearly facilitates #2, Obtaining access to data stored in databases; and #4, Providing a gentle introduction to SQL. It does allow #3, Searching and extracting specific data records of interest, but doesn't really support readers/ users to think well about what might be interesting, or how to frame questions in large complex data sets. And it's not at all clear how the tool accomplishes Objective #1, Familiarizing students with databases and exploring the utility of databases for efficient data storage beyond merely working with a database — the GUI doesn't seem to address any back-end issues let alone explore different possibilities for structuring data.

The authors' implicit instructional (and research) approach seems to be very procedural, and this undermines the possibility of true exploration and supporting deeper conceptual understanding. For example, just above heading 4, they say "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." Is the goal just getting the answer the instructor was looking for? Are there things that students could learn *about* sample size, or relationships among variables, by exploring more on their own? These are not pursued.

One good aspect of the paper is that they actually test and report the results of an empirical study of usability by a mix of undergraduate and graduate statistics students rather than just making a claim about their GUI without any evidence to support it. However, there are also weaknesses in the usability study and the report on the study.

The authors claim that the study assessed student understanding, but there's no real evidence of this beyond performing correctly – e.g., there were no interviews or open-ended feedback sections of the study to actually check with students about what they thought was going on with the GUI, their impressions of what works well about it and/ or what's confusing and could be improved. Looking through the questions in Table 2 confirms this reader's sense about the limited claims that should be made from this study: the first 3 questions are about reading information from the GUI correctly; the next 4 are about reading more than one piece of information and comparing them; and the last 3 (the table is missing #8) are about following instructions with a bit of interpretation – but not about actually understanding SQL. (I'm also confused about how more undergrads could have gotten #11 correct than #10, since #11 depends on #10.)

The paper doesn't report on how errors were coded (including some examples so we can understand your scheme would help) and this seems essential for us to make sense of the findings and what they mean.

Many of the findings are reported in the form of graphs that are essentially unreadable because the type size is outrageously too small — I can just barely read these if I view them at 200%; 400% makes them easily readable, but then the graphs only barely fit on my screen. Furthermore, the graphs don't have a consistent scale so they seem to suggest

the same proportions of students get each question correct when there are, in fact, large differences between them.

While the graphs show undergraduate and graduate responses in different colors, it's not clear if there are differences between these groups on any of these ratings – did you conduct a statistical test (if so, report it). And I'm skeptical about the one difference you do report, that "graduate students show a more clear understanding of the importance of database knowledge as compared to the undergraduate students." To me, what you're noting is that graduates *agree* with you about the importance of this, but we don't know anything about their understanding, and we don't know if they're agreeing because of your GUI or other experiences or cultural expectations, etc. While it does seem from the graphs that there's a pre- post- difference in students' assessment of the difficult of doing the SQL task, again this finding would benefit from a more rigorous analysis/ statistical test.

The suggested next steps about joining data across several tables, allowing stratified sampling, and allowing random sampling sound like important and good directions.

Finally, there are a number of errors in the copy – missing words, wrong words, mis-spellings, etc. I would include specifics, but I believe this paper needs a more thorough revision before it could be published, so such copy-edits seem premature.

In general, the GUI only addresses one small aspect of problems associated with working with large complex data sets, and doesn't seem a very innovative approach to doing so. If its primary goal is to teach potential data analysts how to construct SQL queries, it does that a bit more effectively, but could be improved by providing a larger window in which to review the generated SQL query all at once, or by providing real-time links between changes in the interactive GUI parameters and the query text. As it currently stands, it doesn't seem very compelling.

7.4. Response to Review 2

There seems to be some confusion with communication with the reviewer as the paper was submitted as a Technological Innovation as opposed to a Position paper. The reviewer says they don't see how we achieve Objective 1 (Familiarizing students with databases and exploring the utility of databases for efficient data storage) because we don't address any back-end issues or explore different possibilities for structuring data. This is true - the GUI doesn't teach the user about the back end of databases. This, however, isn't the point of the GUI and wasn't what Objective 1 was trying to get across. Now we recognize that without JOIN statements being implemented at this time that the efficiency provided by using a normalized form for the database isn't as obvious; however, we still provide the user with the ability to explore the database and understand the efficiency provided even if we don't directly point it out.

The reviewer goes on to note many perceived problems they have with the usability study. Some of these are noted and improvements can be made in the future if another study is implemented. They mention the type size of the graph fonts and this will be improved if and when the paper is resubmitted.

The reviewer later mentions doing a statistical test on the data collected in the usability study. This wouldn't necessarily be appropriate since the data wasn't a random sample of any kind. We can provide descriptive statistics and one could make an argument for conducting a formal test for differences between undergrads and graduate students but that really isn't the point. They mention that it seems like we're merely trying to get the correct answer from the students instead of facilitating learning. This is silly - we need to be quantitative to assess the effectiveness of the GUI. In a classroom setting of course an instructor could provide a slightly different question set and/or exercise to teach the basics.

They don't make a few good suggestions. They suggest providing a larger query window to review the generated query all at once and providing real time links between changes in the subsample tab boxes and the generated query. These are good suggestions. The larger window is implemented and the structure of a good SQL query is much more apparent now. The interactive real time links between changes and the generated code is a work in progress.