# ECE 656 Winter 2019: Project

There is an extremely large quantity of data stored in relational databases. While we know how to do basic transactions on such data (e.g., debit account to make a bill payment, book a flight, *etc.*), there is an immense amount of information implied by that data which is much less obvious. For example, when a person books a flight, we know that person, or someone close to that person, will be in a different location for some period of time. We also know that person has the resources necessary to book such a flight. Depending on where the person is going, it might be a business or personal trip. If we correlate that flight with other data about the person, we may deduce more facts. For example, if the person has booked many such flights to the same general location, but it is not particularly a business location, it may imply a sick relative. Other data may be available allowing confirmation or refutation of such a hypothesis (*e.g.*, there may be data about a parking lot payment near a hospital, which would support the hypothesis). By analyzing the data in a database, we may infer information and knowledge beyond the raw data.

In this project, you are required to analyze a sizable, complex dataset (Yelp, the Lahman Baseball data, the Kaggle UWMadison data) to see what knowledge can be gleaned from it. However, prior to starting an analysis you must ensure that the data is "cleaned." An analysis is only as good as the underlying data. If the data is garbage, the knowledge inferred will be nonsense. This means that we need to know several things:

1. Does the data satisfy basis sanity checks?
2. If we are only looking at a subset of the database, is the sample we are analyzing representative of the database as a whole?
3. Is the data representative of the general population?

The third of these is not something that we can do much about at this data-cleaning stage, but will be a relevant question when it comes to analysis. For the data-cleaning portion of the project, you are required to identify as many forms of consistency checking and sanity checking as you can, and implement queries to determine if there are problems with portions of the data. You are further required to recommend solutions for such situations. Solutions can include ignoring some portion of the data for analysis or adjusting the analysis in order to compensate for the data skew. Since different users of your system might have different requirements and/or different choices when it comes to proposed solutions, you should implement the data-cleaning in such a way that

1. The choices can be made by an end-user.
2. The choices can be undone (*i.e.*, it is possible to revert to the "uncleaned" database, so that a different user can make different choices.)

Having ensured that the sample is suitably clean, you then need to analyze the data. For this, you are seeking knowledge about what the data can tell you. You are not expected to determine all possible knowledge that can be inferred from the data. To the contrary, it is much better to do a detailed and careful study of some particular things than a shallow study of a large number things. For example, considering the Yelp dataset, things to study include the following:

1. Based on the data, which businesses are declining and which are improving in their ratings? (This is particularly relevant if a business has a very large number of ratings, in which case a small number or recent ratings that are substantially different than the long-term average would not necessarily affect that long-term average, but would reflect a change in the current state of the business.)
2. Different users have different ways of evaluating businesses. For example, some will readily rate businesses well, and must be very upset to rate a business badly. Predict what rating a user will give

a business based on how s/he has rated other businesses and how others have rated that business? This is particularly valuable in enabling customized recommendations for Yelp users.
3. Do operation hours affect the rating of a business?
4. Does review length affect how other users perceive a review?

At a minimum, you are required to implement one of the data miniing algorithms discussed in class (*i.e.*, decision-tree classifier, *a priori* algorithm, ...) as a stored procedure.

Finally, it is, of course, not sufficient simply to analyze all of the data and state the correlations. Validation is necessary. A typical approach to validation would be to divide the data in two, at random. Half of the data is used for analysis, to make predictions about what matters. The other half can then be used to validate or refute your hypothesis. You will want to ensure that your split is representative, per the comments on data cleaning above. (This validation approach would not work for suggested study (2) above. Why not? How could you validate your methodology there?)

Summary of requirements:

1. The project must implement a client/server database application
2. The server-side must include the implementation of a non-trivial data-mining algorithm
3. The amount of network traffic must be minimized.
4. The client must allow three basic operations:
   (a) Clean data
   (b) Analyze data
   (c) Validate analysis

The client side of the project can be in any programming language of your choosing, with the necessary SQL code included as part of that connecting to the database server using the appropriate DB connectivity mechanisms for the language. For example, C/C++ would likely use ODBC, Java would use JDBC, Perl would use the DBI module, *etc.* The client code may need to present additional user choice beyond the basic three described above. For example, if you implement the *a priori* data minining algorithm, the client code should at least be able to specify parameters for that algorithm in the form of: what table, what thresholds on parameters, *etc.*

The data-mining algorithm must be implemented primarily using SQL and primarily on the server, and not by means of gathering data from the database to the client and then implementing the algorithm on the client. This is what is meant by "network traffic must be minimized."

Your code is what will be evaluation, and no report is necessary. However, some README may be necessary so that the evaluator knows what is necessary for the purpose of executing your code.