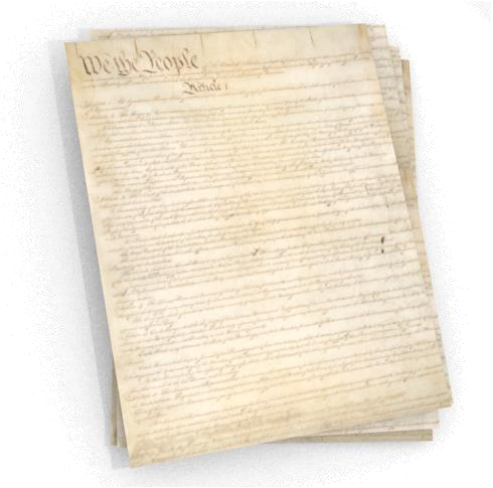
The gap between intent and execution

Data quality is one of those topics that often generates a lot of heat but little light. Technical data users are usually vocal in expressing their opinion on the subject, typically this is a variant of the fact that their data is "high quality" and everyone else's is "poor", but there is much less agreement about exactly what those terms really mean. When I have spare time and a ready supply of refreshment I can easily be persuaded to explore the deeper significance of the term "Quality" [1], but in this context I'll try and avoid that particular temptation. So, how can we get to a simple, and non-debateable, definition of "data quality"?



My personal view is that you can't really claim that any piece of data is "good" without documenting a set of measurable criteria. For me, there has to be some collection of tests that, when they are applied to a particular set of data, provide an indicator of whether the data is "good" (and indeed exactly how good it is). If those tests are not written down then we're talking philosophy and your assessment that "my data is good" is about as useful as a chocolate teapot. If those tests are written down, but subjective, their value is, at best, debatable. To be useful the tests have to be both written down and objective.

But, even this apparently reasonable constraint has the ability to make things challenging. There are some tests that are fairly obvious, for example any Texas well that shows up in the Gulf of Guinea clearly must have something wrong (such as having its position set to 0,0 for example). However, the majority of valuable tests should come from the specialist users of the information, their long experience of mistakes in the data provide the best checks and most relevant tests. Typically those experts would express the intent of each test in a natural language, such as the English "Texas wells should be positioned within Texas". To be truly objective each quality test needs to be able to be automatically "evaluated" by some kind of compute "engine". Which implies that it has to be implemented in a formal language and "executed". For example, we might decide to employ a spreadsheet program to assess quality, in which case the test would be a collection of formulae in cells.

Experience over the last 30 years has clearly demonstrated that these two forms of tests have to **both** be kept, and kept separately from each other. The intent has to be expressed in language the experts can validate and the implementation must be articulated in a directly executable form. Vendors and implementers may claim that their "test definition language" is so easy to understand that the data experts can write it, or that their clever software can parse natural language specifications and interpret them, however in reality such systems don't (yet) work. For the foreseeable future any worthwhile data quality effort needs to keep both the intent and the implementation and use real live people to translate from one to the other. Technically this is easy to implement. In my experience not planning to keep both versions separately from the start will ensure any quality effort will fail.

[1] For example see "Zen and the art of motorcycle maintenance" by Robert Pirsig, a fictional account of a trip across America that discusses, at some considerable length, the meaning of the word "quality".