A Brief History of Data
for the BCS 50th anniversary conference, June 2007
Hugh Darwen
Notes keyed to slides

1. Cover slide
This presentation—mostly a personal memoir—descends from a lecture first given to a British Computer Society audience (the "Formal Aspects of Computing Science" SIG) in December, 1990. In the early 1990s that lecture evolved considerably and found its way around various conferences and British universities. Nowadays its full 4-hour version is given as part of a final-year course for undergraduates at Warwick University, under the title The Askew Wall: SQL and The Relational Model. The slides for that are available at http://www.thethirdmanifesto.com under Presentations (direct url http://www.dcs.warwick.ac.uk/~hugh/TTM/TTM-TheAskewWall-printable.pdf).

2. Past, Present, and Future
The Past is a review of the database scene as I see it having developed since I joined it myself—well, not quite 50 years ago.

The Present is my view of the unfortunate result of that sordid history. The Askew Wall is my name for what I regard as one of the very worst computer languages to have achieved widespread use. I have to explain the term to student audiences, containing, as they do, such a large overseas contingent:

askew, adj. at or to an oblique angle; awry
awry, adj. twisted to one side; distorted, crooked; wrong; perverse
wall, n. ...; a barrier; ...

"Askew Wall" is rather contrived as a pun but it works very well for me as a metaphor, as you can see from those dictionary definitions.

The Future would, if the impossible dream came miraculously true, see The Askew Wall replaced by a language based faithfully on (a) relational theory and (b) generally accepted principles of good language design.

The Third Manifesto (TTM for short) is a paper by the author and C.J. Date, originally published in 1995, inspired by the many mistakes in SQL. It sets out a series of prescriptions that must be followed if a language is to be a faithful implementation of "the relational model of data" (as its inventor, E.F. Codd called it back in 1970). And a series of proscriptions (things that must not be included) that are really consequences of the prescriptions. And a series of Very Strong Suggestions. The paper is explained at length and in detail in a book, Databases, Types, and The Relational Model (Addison-Wesley, 2006) by the same authors.

3. A Brief History of Data (Part 1) and
4. A Brief History of Data (Part 2)
In the days of cards and tapes, all files had to be accessed sequentially, starting with the first record and then repeatedly moving on to the next one until the last one had been processed. This involved the coding of loops in application programs, nested loops when more than one file was needed, as was usually the case. The algorithms involved were quite tricky and a common source of bugs in the application programs.
The advent of disks allowed us to dispense with some of the loops but introduced the equally trap-
laden process of pointer chasing. Disks also raised the possibility of making an organisation's data a shared resource for concurrent access by a possibly diverse community.

Codd recognized both the opportunity and the dangers. He sought a foundation for technology that would allow application programs working with what were later to be called databases to avoid both the use of loops and the use of pointers.

The line for 1975 mentions two languages, ISBL and SQL. ISBL was a brilliant piece of work done by a small research team in the United Kingdom. A notable omission from Codd's papers was any precise definition of a computer language that would materialize his idea. The ISBL team made good this omission, faithfully adhering to Codd's model. The System R team working in the USA had a slightly different objective: to show that Codd's idea could be implemented efficiently enough for commercial use even in the "on-line transaction processing" (OLTP) environment. But the System R team of skilled "engine-room" workers didn't actually understand Codd's idea as well as the ISBL team and weren't very good language designers either.

It isn't always the good work that achieves high visibility and acclaim.

As for the history of data since 1980, the slides say it all really. Not everybody might see much significance in the 1995 event but I did say this was a personal view of the history!

5. A Brief History of Me (Part 1) and

6. A Brief History of Me (Part 2)

Terminal Business System was an early DBMS, before that term had even been coined. Working on it, as a fairly junior programmer to begin with, I learned about the problems of database management and the typical solutions that were known about in those days. But Terminal Business System applications had to use loops and chase pointers. We included a couple of simple scripting languages for maintaining the database and generating reports, but in each case a script could access no more than one file and required the file to have records of uniform format. These scripting languages were called, prosaically, File Maintenance and Report Writer. Users complained about their limitations. File Maintenance scripts couldn't include checks for consistency with other files in the database, or even among different records in the same file; and Report Writer scripts couldn't combine information from several files.

So that's why my attendance on Chris Date's course in 1972 was a "personal watershed". His account of the new idea from E.F. Codd gave us the ideal solution to the very problems our customers were begging us to solve. And I had been working, in vain, seeking possible solutions that very year. The real solution that now presented itself as far as Report Writer was concerned was beautifully simple: just replace the input file by a relational query result and leave everything else the same.

Six years later it was time to develop and new DBMS and by now the relational model had attracted enough attention to make it the obvious choice. We looked around the research world to see what it had made of Codd's idea and discovered ISBL, which not only solved all of the problems in language design that we had been grappling with but did so in a delightfully simple way, fully embracing the use of attribute names. We also discovered SQL but quickly rejected it as being not only unfaithful to the relational model but also extremely baroque and unconventional. And at that time relationally incomplete. I assured the executive manager who was to approve our funding that SQL would never catch on!

7. The Perversity of SQL

The first SQL query on this slide was once set as an exercise for students of The Open University, who were asked to explain in ordinary English what the query is really asking for. Even the
students who did understand SQL "subqueries" struggled with it, because of the perverse order in which the main "less than" comparison is written. And so did I, a tutor to some of these students. Anyway, the example set in motion a long sequence of events that was to culminate in a change to the SQL international standard (!) as I relate in the lecture.

Which is easier to understand: cities where 4 is greater than the number of other cities with bigger populations, or cities where the number of other cities with bigger populations is less than 4? Is a mile less than the longest distance you can run non-stop, or can you run for more than a mile, non-stop? Is 100 less than the best score Shane Warne has ever made at batting in cricket? Or has Shane Warne ever scored more than 100?

It's only a psychological issue, of course, but the example illustrates one of the many examples of poor language design by the research team at IBM that invented SQL back in the 1970s: the revised query in the lower half of the slide was illegal in their language, even though it merely converts a couple of examples of \( a > b \) into \( b < a \). And the error still survives in many SQL implementations to this day, even though it was corrected in the international standard as long ago as 1992 ...

... However, shortly after the publication of SQL:1992, a proposal was accepted to add to the Technical Corrigendum for SQL:1992 a clause to the effect that an implementation seeking conformance to just the "core" level of conformance would be permitted to keep that restriction on subqueries in comparisons. It was that proposal, approved at the committee meeting in Munich, Germany, in January 1993, that so exasperated the present author as to cause him to sit down in his hotel room with pencil and paper over one weekend, produce the first draft of *The Third Manifesto*, and fax it to Chris Date.

### 8. The Career of a Coauthor

Apologies for referencing nothing but my own work and Chris Date's, but not a lot of people have written about this particular subject, SQL and the relational model, in this way. Anyway, the real purpose of this slide is not what it seems, as becomes clear during the presentation.

### 9. Sonnet 122

The timeless bard, William Shakespeare, wrote his 122nd sonnet not to some Dark Lady, not to the Earl of Southampton, but to Ted Codd. I reproduce it here in full, with one tiny orthographical correction by me—a matter of a capital letter.

```
Thy gift, thy tables, are within my brain
Full character'd with lasting memory,
Which shall above that idle rank remain
Beyond all Date, even to eternity;
Or at the least so long as brain and heart
Have faculty by nature to subsist,
Till each to razed oblivion yield his part
Of thee, thy record never can be missed.
That poor retention could not so much hold,
Nor need I tallies thy dear love to score;
Therefore to give them from me was I bold,
To trust those tables that receive thee more.
    To keep an adjunct to remember thee
    Were to import forgetfulness in me.
```

### 10. The Fatal Flaws of SQL

This is just a list of the topics that I deal with in the detailed (4-hour) version of this lecture. Each flaw represents a major departure from relational theory. The consequences of such departures
need to be examined and understood, and their severity needs to be assessed. The term "fatal flaw" was used by Ted Codd, but he didn't mention all the ones I do.

11. The Hunting of The Snark

I include this slide just for fun. The duplicate row phenomenon is only about the second worst of SQL's flaws and I have no other good reason for singling out this particular flaw in preference to, say, NULL, anonymous columns, or duplicate column names.

The 19th century logician, Charles Lutwidge Dodgson, also known as Lewis Carroll, the writer of nonsense, joked about SQL's so-called "duplicate row phenomenon" a century in advance. Here are the opening lines of his famous epic poem:

"Just the place for a snark!" the bellman cried
As he landed his crew with care,
Lifting each man on the tip of the tide,
By a finger entwined in his hair.

"Just the place for a snark! I have said it twice—
That alone should encourage the crew.
Just the place for a snark! I have said it thrice.
What I tell you three times is true."

Or, as Ted Codd put it, "If something is true, saying it twice doesn't make it any truer."

12. A Guiding Light

Date and I were both rather struck by this saying of the famous 20th century philosopher Ludwig Wittgenstein when it was drawn to our attention by my friend, Adrian Larner. It took us several years to track it down. It appears that Wittgenstein happened to say it in the presence of his contemporary at Cambridge University, Peter Geach, with whom Larner had been acquainted when he was a student at that university in the 1960s.

13. The Dream Database Language

… is any that conforms to The Third Manifesto, which is really only intended as a clarification of Codd's 1970 work in the light of the various errors and misunderstandings that have arisen over the years (manifested mainly in SQL). The point about user-defined types and operators was highly relevant in the 1990s, when "object-relational" was the theme of the decade and more. "Chivalry" of course refers to the ease of use that accrues from faithful adherence to the model—and to the generally accepted principles of good language design.

14. D

Sorry for the hubris. Actually, we use D as a generic name for any language that conforms to The Third Manifesto. Some people like to include D in the language name, as we did in Tutorial D. The term Industrial D is sometimes used for the one that doesn't exist (yet).

15. Projects based on TTM

One that isn't mentioned on the slide is D♭ (D-flat, to go with Microsoft's C#!)

16. Conceptual Integrity

Principle #7 because there are six others in the full version.

Brooks had been working for IBM as the manager in charge of the development of the huge, baroque operating system, prosaically called OS, that was used on the big IBM mainframes of the 1960s and 1970s. His famous book documenting his experiences and conclusions was entitled The Mythical Man-month. Note carefully the italicization of the word the in the quotation. I heard him
give it the same emphasis orally when talking about conceptual integrity in an after-dinner speech I attended in 1995.

17. Reims Cathedral

This is the example used by Fred Brooks in that book, *The Mythical Man-month*. The cathedral took perhaps 100 or more years to build in the middle ages, but the builders stuck slavishly to the original concept and this was the result. I was in that part of France in 1999, hoping to witness a total eclipse of the sun. The eclipse was a wash-out but it was good to pop into the cathedral to check out Brooks's analogy.