communication +1

Volume 1 *Futures of Communication* Article 4

8-29-2012

From Punched Cards to "Big Data": A Social History of Database Populism

Kevin Driscoll
University of Southern California, kedrisco@usc.edu

Abstract
Since the diffusion of the punched card tabulator following the 1890 U.S. Census, mass-scale information processing has been alternately a site of opportunity, ambivalence and fear in the American imagination. While large bureaucracies have tended to deploy database technology toward purposes of surveillance and control, the rise of personal computing made databases accessible to individuals and small businesses for the first time. Today, the massive collection of trace communication data by public and private institutions has renewed popular anxiety about the role of the database in society. This essay traces the social history of database technology across three periods that represent significant changes in the accessibility and infrastructure of information processing systems. Although many proposed uses of "big data" seem to threaten individual privacy, a largely-forgotten database populism from the 1970s and 1980s suggests that a reclamation of small-scale data processing might lead to sharper popular critique in the future.

Keywords
data, database, big data, code, platform, software, history of technology

Cover Page Footnote
This research was supported in part by a USC Annenberg Graduate Fellowship.

For over a century, the large institutional database has been as a symbol of domination, surveillance, and control in U.S. popular culture. From military conscription to credit scoring, database technology is imagined to serve a dehumanizing function, transforming the individual subject into a bloodless set of statistics. Today, the massive, unregulated collection and algorithmic manipulation of trace internet communication data by public and private institutions has renewed popular fear of the database and invites a historical review of the changing role of the database in society.

Although early users of database technology were predominantly large institutions, the database was also a key technology in the populist vision of personal computing generated by microcomputer fans, researchers, hobbyists, and entrepreneurs in the 1970s and 1980s. Informed by science fiction sensitive to the authoritarian use of database technology, these personal computing advocates hoped that experience with small database systems might sharpen popular critique of mass-scale information processing efforts. As database design receded from the desktop in the 1990s, however, the populist promises were largely forgotten and the database became an exclusively institutional technology once again.

Today, the development of new database technologies is driven by the demands of extremely large data sets, especially those produced by highly-centralized web services such as Google and Facebook—a cross-cutting field of research colloquially termed "big data." While these new systems inherit characteristics of both the institutional and personal approaches, their infrastructures are often spread across thousands of geographically dispersed machines and require users and programmers to develop new habits of mind in order to effectively communicate with them. Computer scientists are just beginning to tackle the theoretical and methodological problems present in such "big data" projects, yet unfounded claims about the explanatory and predictive power of large data sets are increasingly common in the discourses of both public and private institutions.

A social history of database technology situates the web's massive databases among more than a century of mass-scale information processing systems. Evidence of popular anxiety recurs throughout this history and indicates that non-specialists often struggle to apprehend the limits of database technology and may alternately over- and under-estimate the extent of mass data collection and the types of analytic outcomes that are possible. Meanwhile, the cautious optimism of the microcomputer era points to a latent database populism that may yet be revived should users grow sufficiently frustrated by the lack of transparency among large, data-driven institutions.

Databases and the study of algorithmic culture

"Database" is a term of relatively recent origin, first appearing in academic papers during the 1960s.[1] Computer scientists are careful to distinguish the database, which deals strictly with data storage, from the database management system (or DBMS) that also includes tools for organizing, analyzing, and manipulating those data. The colloquial use of "database," however, tends to blur this distinction. To facilitate historical comparison, "database" may be defined broadly as a system for organizing information according to a generic model and storing it in a material form. Referring to systems from before the 1960s as "databases" is apocryphal but helps to emphasize the continuity among similar terms such as "information processing" and "data management."

In the emerging scholarship concerning the role of algorithms in online communication, databases are often implicated but rarely of principle concern. This subordinate position may be due to the ambiguous relationship between algorithm and database. Whereas an algorithm, implemented in running code, is self-evidently active, a database appears to serve a largely passive role as the storehouse of information. In such an arrangement, the database may be understood as a function or component of the algorithm without weakening the overall analysis. For example, John Cheney-Lippold usefully offers the term "algorithmic identity" to describe the product of data-mining software as it tracks users across the web and attempts to statistically determine individual characteristics such as sex, gender, and race.[2] "Code and algorithm," he writes," are the engines behind such inference."[3] Implicit in this metaphor is a database—or, more likely, a network of databases—from which the engine (code) draws its fuel (data.) As Cheney-Lippold's analysis makes clear, algorithmic identity arises out of the interaction between code and data, but without an available collection of user data, the code remains inert and algorithmic identity construction is impossible.

---

[1] "ACM SIGMOD Conference," The DBLP Computer Science Bibliography, accessed June 12, 2012, http://www.informatik.uni-trier.de/~ley/db/conf/sigmod/. The early proceedings of the ACM SIGFIDET and SIGMOD conferences reveal the term "database" in negotiation among computer scientists. In some papers, "data" and "base" are two separate words; in others they are hyphenated. M. Lynne Neufeld and Martha Cornog, "Database History: From Dinosaurs to Compact Discs," Journal of the American Society for Information Science 37, 4 (1986): 185, doi:10.1002/(SICI)1097-4571(198607)37:4<183::AID-ASI2>3.0.CO;2-W. According to Neufeld and Cornog, a standard spelling was not agreed upon by information scientists until 1979.

[2] John Cheney-Lippold, "A New Algorithmic Identity Soft Biopolitics and the Modulation of Control," *Theory, Culture & Society*, 28, 6 (2011): 164-181, accessed June 12, 2012, doi:10.1177/0263276411424420.

[3] Ibid., 165.

Designing an algorithm always involves consideration for the structure of the data on which the algorithm will act. The efficiency of an algorithm is measured in terms of the number of steps (i.e., processor cycles) required to manipulate the given data structures. Seemingly simple tasks such as adding a new element to a list can become quite taxing if the list is sorted or if all the elements in the list must be unique. For a low-tech example, imagine arranging books on a bookshelf. If the books are sorted alphabetically, fewer steps are required to locate a particular title but more are needed to add a new book to the shelf. On the other hand, if the books are randomly placed on the shelf, it will be easier to add a new book but harder to find a particular one. For this reason, introductory courses on algorithms invariably dedicate considerable space to the study of data structures. The first of Donald Knuth's canonical computer science textbooks is titled, *Fundamental Algorithms* but fully half of the book is dedicated to the subject of "information structures."[4] Indeed, "garbage in, garbage out," a tongue-in-cheek programmer aphorism,[5] underscores the interdependence of algorithm and data. Even the most elegantly composed algorithm will fail to produce satisfactory results in the absence of appropriately structured data.[6]

The entanglement of algorithm and data is especially challenging for studies of highly-centralized, privately-owned communication systems. The algorithms that drive Google search results, Amazon recommendations, and Twitter trends remain closely-guarded trade secrets. Not only are the mechanics of these algorithms obscured from the user, they are also constantly in flux. Given an identical set of inputs, there is no guarantee that these black box systems will produce the same output twice. Furthermore, any effort to recreate or reverse engineer these algorithms is constrained by unequal access to input data. Even relatively generous institutions such as Twitter do not permit outsiders to access their databases directly.

In a recent essay, Tarleton Gillespie examined the recurring charge among Occupy Wall Street supporters that Twitter censors its "Trending Topics" system.[7] Despite significant discussion of the Occupy movement on Twitter during the fall of 2011, the unifying term "#occupywallstreet" never appeared in

---

[4] Donald Knuth, *The Art of Computer Programming, Volume 1: Fundamental Algorithms* (Reading, MA: Addison-Wesley, 1973).

[5] Eric S. Raymond, ed., "GIGO," *The Jargon File*, accessed July 21, 2012, http://catb.org/jargon/html/G/GIGO.html.

[6] In a related study of this "symbiotic relationship," Lev Manovich explored the tension between unordered objects in a database and the production of narrative in new media. "Database as Symbolic Form," *Convergence* 5, 2 (1999): 89-99, doi:10.1177/135485659900500206.

[7] Tarleton Gillespie, "Can an Algorithm be Wrong?," *Limn*, 2, (2012), accessed June 12, 2012, http://limn.it/can-an-algorithm-be-wrong/.

Twitter's list of Trending Topics. Frustrated movement supporters concluded that censorship, rather than design or malfunction, best explained the absence. Gillespie notes that charges of censorship assume that the underlying communication system is otherwise functioning justly. Therefore, when Occupy supporters accused Twitter of censorship, they revealed a certain faith in the fairness of the opaque mechanics of the Trending Topics system.

One reason that users might turn to human intervention to explain system failure is that they lack the necessary vocabulary to generate or evaluate possible algorithmic explanations. Without the language to describe an alternative to the failing Trending Topics algorithm, users overemphasized the role that the algorithm's designers and maintainers played in its moment-to-moment behavior.[8] The inability of Occupy protesters to articulate their frustration with the Trending Topics algorithm exemplifies one type of fear that has long accompanied mass-scale information processing systems. As users entrust opaque institutions such as Twitter to responsibly steward their intimate communication, they make themselves vulnerable to a remote technical system that exceeds their discursive capacity to express expectations and grievances.

Resolving the conflict between users and institutions like Twitter is difficult because the ethical stakes remain unclear. Is Twitter ethically bound to explain its internal algorithms and data structures in a language that its users can understand? Conversely, are users ethically bound to learn to speak the language of algorithms and data structures already at work within Twitter? Although social network sites seem unlikely to reveal the details of their internal mechanics, recent "code literacy" projects indicate that some otherwise non-technical users are pursuing the core competencies necessary to critically engage with systems like Twitter at the level of algorithm and database.

Facing apparent inertia on the part of large institutions to become more transparent, a rising number of voices argue for a technical awakening among non-specialists. Recalling the urgent cover of Ted Nelson's *Computer Lib*, which read, "You can and must understand computers now!,"[9] these critics advocate a kind of rugged digital individualism in which users must acquire and deploy

[8] Although the specific algorithms that produce Trending Topics remain trade secrets, a combination of defensive PR by Twitter and clever post-hoc analysis by third-party researchers indicates that trends are not based on simple frequency but a more complex analysis of a topic's circulation. See: Twitter, "To Trend or Not to Trend…," *Twitter Blog*, December 8, 2010, accessed August 22, 2012, http://blog.twitter.com/2010/12/to-trend-or-not-to-trend.html; Gilad Lotan, "Data Reveals That "Occupying" Twitter Trending Topics is Harder Than it Looks!," *SocialFlow Blog*, October 12, 2011, accessed August 22, 2012, http://blog.socialflow.com/post/7120244374/data-reveals-that-occupying-twitter-trending-topics-is-harder-than-it-looks.

[9] Theodor H. Nelson, *Computer Lib*, (Self-published, 1974).

technical knowledge defensively to avoid exploitation.[10] Eli Pariser, an outspoken critic of personalized search results and other algorithmic echo chambers, urges users to "vary [their] online routines, rather than returning to the same sites every day."[11] In a presentation about the threat of pervasive data-mining to individual privacy, Gary Kovacs of Mozilla Corporation implored the audience at TED, "We are being watched, it is time for us to watch the watchers."[12] And, perhaps most forcefully, Douglas Rushkoff links code competency to self-determination, "If you are not a programmer, you are one of the programmed."[13]

As compelling as the rugged digital individualist position can be, the pathways to action are not always clear, leaving some converts with an uneasy feeling that they are abdicating a duty they remain ill-equipped to manage. A startup called Codecademy briefly brought this anxiety to light with their "Code Year" project in late 2011. More than two hundred thousand people, including New York City Mayor Michael Bloomberg, signed a pledge to "learn to code in 2012."[14] On blogs and messageboards, programmers expressed mixed feelings about the initiative. Most believed that exposure to programming and code would be helpful for understanding the role of software in society, but they balked at the notion that Code Year's weekly exercises could teach novices how to write useful software. In a blog post titled "Please don't learn to code," Jeff Atwood wrote, "I would no more urge everyone to learn programming than I would urge everyone to learn plumbing."[15]

As the founder of StackOverflow, a popular site for I.T. professionals, Atwood's blog is widely read by programmers. Of the 217 responses to Atwood's

---

[10] Thomas Streeter thoroughly explores the various forms of individualism that have accompanied cultures of networked computing over the past half-century in *The Net Effect: Romanticism, Capitalism, and the Internet* (New York: New York University Press, 2011).

[11] Maria Popova, "The Filter Bubble: Algorithm vs. Curator & the Value of Serendipity," *Brain Pickings*, May 12, 2011, accessed August 22, 2012, http://www.brainpickings.org/index.php/2011/05/12/the-filter-bubble/. See also: Eli Pariser, *The Filter Bubble: What the Internet Is Hiding from You* (New York: Penguin, 2011); Eli Pariser, "10 Ways to Pop Your Filter Bubble," *The Filter Bubble*, accessed June 12, 2012, http://www.thefilterbubble.com/10-things-you-can-do.

[12] Gary Kovacs, "Gary Kovacs: Tracking the Trackers," *TED*, accessed June 12, 2012, http://www.ted.com/talks/gary_kovacs_tracking_the_trackers.html. See also: http://www.mozilla.org/en-US/collusion/.

[13] Douglas Rushkoff, *Program or Be Programmed: Ten Commands for a Digital Age* (Soft Skull Press, 2011).

[14] Laurie Segall, "Code Year Draws 200,000 aspiring programmers," *CNN Money*, January 6, 2012, accessed June 12, 2012, http://money.cnn.com/2012/01/06/technology/code_year/index.htm. See also: http://codeyear.com/.

[15] Jeff Atwood, "Please Don't Learn to Code," *Coding Horror*, May 15, 2012, accessed June 12, 2012, http://www.codinghorror.com/blog/2012/05/please-dont-learn-to-code.html.

post, a pseudonymous comment by a reader called Elective-C stands out from the rest. Joining the self-sufficiency of the digital individualist to the domain expertise of a software engineer, Elective-C argues that power and control are found in data structures, not algorithms:

> I agree [with Atwood] 50%. Programming is often taught as a combination of two things: algorithms and data structures. Algorithm instruction should be left to those of us who revel in them. However[,] everyone needs to understand data structures... and all too few people do[.] So, algorithms are optional, but in the future everyone must be data literate... or their lives will be destroyed by those who are.[16]

Elective-C makes a strong case but the voices of "data literacy" advocates are easily overwhelmed by the rousing chorus demanding "code literacy." One explanation for this imbalance is that data structures and database technologies serve as the infrastructures on which algorithm-driven systems depend. Consistent with Susan Leigh Star's observation that infrastructure can seem "mundane to the point of boredom"[17] and be easily overlooked, the database is woefully underexamined in the sociology of computing. Although laboratory studies and other workplace ethnographies frequently consider specific databases encountered in the field, little scholarly attention has been dedicated to the social function of database technology writ large.

To address this gap, this essay presents a history of the database in the U.S. by looking at three periods marked by changes in the accessibility and infrastructure of database technologies. The first period begins in the late-19th century with the development of mass-scale information processing projects and the electro-mechanical punched card systems that made them possible. Although these early machines were gradually replaced by programmable computers in the 1950s and 1960s, the organizational logic embedded in such systems persisted more or less unchanged until the 1970s. The second period is marked by the rise of database populism and the increasing availability of microcomputers in the late-1970s. Implementations of the relational data model enabled the production of more accessible interfaces for non-specialists and large institutional databases were increasingly accompanied by small personal databases built by individuals and stored on microcomputers. In the third period, however, small personal databases receded from the desktop with the increasing sophistication of

[16] Elective-C, May 15, 2012 (3:04 a.m.), comment on Jeff Atwood, "Please Don't Learn to Code," *Coding Horror*, May 15, 2012, http://www.codinghorror.com/blog/2012/05/please-dont-learn-to-code.html.

[17] Susan Leigh Star, "The ethnography of infrastructure," *American Behavioral Scientist*, 43, 3 (1999): 377, accessed June 12, 2012, doi:10.1177/00027649921955326.

spreadsheet software and the diffusion of internet access. In the early 21st century, the demanding task of tracking millions of users through highly-centralized communication systems such as Facebook brought about new approaches to database design that departed significantly from the previous four decades. These new database systems traded the accessibility and stable structure of the relational model for the capacity to handle an extremely large flow of irregular data spread out among thousands of servers in a network. This three-part narrative is not a comprehensive chronicle of database theory and technology nor does it address the political economy of most database management systems. Rather, it should provide a foundation for future inquiry into the social impact of mass-scale information processing.

The evidence supporting this project is drawn from an unusual archive of hobbyist periodicals, popular histories, computer science publications, and secondary historical sources. This trans-disciplinary literature reflects the tendency of databases to cut across conventional social boundaries. In different contexts, database technologies have been understood as media, infrastructures, products, and tools. Historians of information science, computing, and the internet, to whom this essay is indebted, have illuminated various features of the database through the lenses of their home disciplines. The periodization constructed here is an attempt to negotiate and synthesize these different approaches toward a robust, multi-perspectival understanding of databases as they have been designed, populated, queried, critiqued, and destroyed.

Early machinery for mass-scale information processing

At the end of the 19th century, public and private bureaucratic institutions were growing increasingly large and geographically dispersed, creating a rising demand for new systems to manage information. In his thorough account of the period, James Beniger identifies numerous technologies—the paper clip, the manila folder, the filing cabinet—that quickly became part of a standard repertoire of office supplies.[18] In addition to these small tools, more complex systems were devised for storing, organizing, and analyzing information at a mass scale. The resulting electro-mechanical information processing systems were almost wholly inaccessible to the non-specialist. They were expensive to build, narrowly designed for specific tasks, and difficult to modify. These proto-databases were built to support the work of very large bureaucracies, such as governments and

[18] James R. Beniger, *The Control Revolution* (Cambridge: Harvard University Press, 1986). These office technologies also serve as the metaphorical repertoire from which the standard graphical computer interface is assembled.

multidivisional corporations, and only such institutions could afford to commission them.

Despite the rarity of these systems, most people living in the U.S. at the turn of the 20th century would have had an occasion to encounter mass-scale information-processing during the course of their lives. Unfortunately, these encounters were likely not terribly pleasant for most. Early applications of information processing by the state tended to use information for "the active control of individuals."[19] Fingerprinting for the incarcerated, psychological screening for draft inductees, and income tax for working people were all semi-automated information processing systems in place in the U.S. before 1920. The first truly mass-scale state information processing project to be mechanized, however, was the eleventh U.S. Census beginning in 1890.

The 1880 census involved a significant increase in scope and scale from previous efforts, and was, consequently, still incomplete after nine years.[20] Nevertheless, the 1890 census was scheduled to outdo its predecessor.[21] Not only was the lengthy 1880 questionnaire preserved in full but additional items were planned for inclusion.[22] Anticipating the volume of data that would soon be coming in from around the country, the Census office solicited suggestions for an information processing system to replace the unwieldy handwritten paper rolls then in use. Herman Hollerith, a census employee, submitted the subject of his PhD thesis: an electro-mechanical machine for tabulating information recorded onto punched cards.

Drawing inspiration from a railway ticketing system,[23] Hollerith proposed that each record in the 1890 census be stored on an individual punched card. Breaking up the dataset into materially-discrete records offered two distinct advantages over the paper roll system in terms of accessibility. First, a clearly organized set of punched cards provided random access to any single record

[19] Ibid., 408.

[20] Ibid., 409-411. The 1880 census gathered information on more than 215 different subject areas. Census workers in the field shipped their records to a central office where clerks transferred the individual responses onto long rolls of paper by hand. This bureaucracy was easily overwhelmed and the 1880 data were still being transcribed in 1889 when the next census collection was set to begin.

[21] Robert L. Dorman, "The Creation and Destruction of the 1890 Federal Census," *The American Archivist* , 71, 2 (2008): 357.

[22] Ibid., 356.

[23] Beniger, *Control Revolution*, 410-411. While working on a project unrelated to the census, Hollerith observed an unusual anti-fraud system in use among railroad conductors. At the point of purchase, the conductor would manually punch holes in certain areas of the ticket to indicate features of the purchaser's appearance: eye color, hair color, facial hair, etc. Each ticket was thus uniquely tied to its original buyer.

whereas the paper roll needed to be parsed serially from the beginning in order to reach a given record. Second, the machine-readable cards enabled census analysts to more efficiently issue ad hoc queries against the data. Instead of manually checking every row on a paper roll to see if it matched a set of given criteria, the analyst could use a sorting machine to rapidly sift through a stack of cards and produce a desired subset. Over the next century, the need for random access and an expressive ad hoc query method continued to characterize the development of information processing systems.

Hollerith's machines facilitated a considerably faster tabulation of the 1890 census and between 12.5 and 15 million individual records were processed in fewer than two years.[24] Unfortunately, the 1890 census was very much a statistician's project and little consideration was given to the value that the collected data might provide to future historians or genealogists. Poor archival practices lead to a terrible loss when the only copies of the handwritten records were destroyed in a fire in 1921. This avoidable calamity highlights one of the dangers of ignoring the differences among various data management systems. To many members of the 1890 census team, the production of punched cards from handwritten forms was taken for granted as a straight-forward transfer—rather than a transformation—of the data. In Dorman's words, "the punched cards became, in effect, not only a copy of the information, but its principal medium."[25] This elision of structural difference obscured the affordances and constraints of each medium. Whereas a machine could tabulate a stack of punched cards quite rapidly, considerable labor was required for a human reader to interpret even a single card. Due to a short-sighted design decision, each of the millions of cards was stamped with a unique number corresponding to a family name. The index of card numbers and family names thus represented a significant vulnerability. Should the index be lost or destroyed, there would be no way to link cards to families.

In spite of its archival weaknesses, the Hollerith-driven census was a highly-visible success and punched card tabulators were soon adopted by bureaucracies throughout the federal government.[26] Numerous taken-for-granted government functions, such as library catalogs and automobile registration, only became possible alongside an infrastructure that could support mass-scale information processing.[27] In the 1930s, nationwide progressive social programs resulted in a need for data processing capacity at unprecedented speed and scale.

[24] Dorman, "Federal Census," 366.
[25] Ibid., 358.
[26] Through a series of mergers and re-namings, the Hollerith corporation became IBM in the early 20th century.
[27] Beniger, *Control Revolution*, 419-420.

In terms of raw throughput, the Social Security system monitored twenty-six million people and processed approximately 500,000 punched cards per day during its first years.[28] Private industry was similarly eager to apply massive data processing infrastructure to its internal functions. The growth of the insurance industry depended on the ability to run statistical analyses on large populations; telephone and utilities companies used tabulators to bill large numbers of customers for small amounts; and manufacturing firms developed real-time accounting methods for tracking fluctuating costs and sales across large multidivisional corporate structures.[29]

For workers in these large institutions, the advent of mass-scale information processing presented new employment opportunities. During the first half of the 20th century, a steadily increasing proportion of the overall working population was employed to do technical labor within a bureaucracy. In addition to novel occupations like stenographer, typist, bookkeeper, and cashier, a significant number of employees were required to maintain and operate electro-mechanical data processing systems.[30] Although the idiosyncratic, specialized designs of these early machines may have limited the transferability of their skills, operators of the data processing machines likely enjoyed a greater degree of job security than many of their peers. Early-adopting institutions tended to keep their bespoke systems in service for multiple decades. Prudential Life Insurance, one of Hollerith's first customers, continued to use its punched card system for over forty years before replacing it with a general-purpose computer system.[31] Even then, early commercial computers did not constitute a radical departure from the electro-mechanical machines they replaced. The IBM 650, for example, was designed to ease institutional transition during the 1950s and continued to read and write the same punched cards as its mechanical predecessors.[32]

Fear and anxiety marked much of the popular response to database technology in this early period. For some, the punched card provided material evidence of the dehumanizing power of the bureaucracy. Rather than communicate with individuals, in all their messiness, large bureaucracies preferred inscrutable slips of perforated paper. That such systems were used to track taxes, insurance rates, and debt surely contributed to a popular ambivalence regarding the promises of database technology. In addition, just as institutions in the U.S. deployed data processing infrastructure in pursuit of surveillance and

[28] Ibid., 408.
[29] Ibid., 423.
[30] Ibid., 395.
[31] Ibid., 396-397.
[32] José-Marie Griffiths and Donald W. King, "US Information Retrieval System Evolution and Evaluation (1945-1975)," *IEEE Annals of the History of Computing* 24, 3 (2002): 42.

control, so were less democratic governments eager to adopt mass-scale information processing techniques. Punched card tabulators (and, later, programmable computers) were essential tools for central economic planning, a fact that may have linked these machines to totalitarianism, forced migration, and systematic slaughter in the American imagination.

Alternative visions of the database in society

Although general-purpose programmable computers and magnetic storage gradually replaced mechanical tabulators and punched cards during the intervening decades, the role of the database in society remained remarkably stable from 1890 to 1960. With few exceptions, information processing systems enabled large institutions to act upon—and not on behalf of—individuals. For young people in the post-war period, the systematized horrors of the Nazi and Stalinist regimes and the pervasive threat of nuclear war seemed to confirm the anti-human capacity of mass-scale bureaucracy.[33] As the antiwar movement began to take shape across the U.S., draft card burning represented the material destruction, however symbolic, of the state's information processing apparatus, and at many universities, protesters held demonstrations outside of campus computing facilities to disrupt military-funded information processing research.[34] As the caption to a comic strip in *Mad* would later put it, "the punch cards are stacked against you."[35]

---

[33] Fred Turner offers a helpful summary of the Students for a Democratic Society opinion regarding the social outcome of mass-scale information processing: "A highly bureaucratized society whose structures virtually required individuals to become psychologically fragmented and thus capable of atrocious horror." *From Counterculture to Cyberculture: Stewart Brand, the Whole Earth Network, and the Rise of Digital Utopianism* (Chicago: University of Chicago Press, 2006): 34.

[34] Many of the graduate students and faculty working in these facilities were sympathetic to the peace movement. John Markoff indicates that some students chose to quit the Computer Science lab at the Standford Research Institute on moral grounds and M. Mitchell Waldrop notes that other students justified their compromised position with a belief that they were "co-opting" the government by diverting funds that would otherwise go to the manufacture of weaponry. See: John Markoff, *What the Dormouse Said: How the 60s Counterculture Shaped the Personal Computer Industry* (New York: Viking, 2005): 171-174; M. Mitchell Waldrop, *The Dream Machine: J. C. R. Licklider and the Revolution That Made Computing Personal* (New York: Penguin, 2001): 281-282.

[35] Tom Koch and Bob Clark, "We're Losing Our War Against Computers," *Mad*, 171 (1974), reprinted in David H. Ahl, ed., *The Best of Creative Computing: Volume 1* (Morrisstown, NJ: Creative Computing Press, 1976): 122-123.

Research libraries and the field of library science offered an important space in which an alternative vision of data processing could emerge.[36] As H. G. Wells wrote in his 1937 essay on the construction of a "permanent world encyclopaedia,"

> Few people as yet, outside the world of expert librarians and museum curators and so forth, know how manageable well-ordered facts can be made, however multitudinous, and how swiftly and completely even the rarest visions and the most recondite matters can be recalled, once they have been put in place in a well-ordered scheme of reference and reproduction[.] The whole human memory can be, and probably in a short time will be, made accessible to every individual.[37]

Michael Buckland's work on the history of information management offers the most comprehensive account of early systems designed to augment the work of individual researchers.[38] Across Europe and the U.S., proto-hypertext systems re-imagined the research library as a database that might be queried, sifted, and sorted in the same manner as a deck of punched cards.

Although the innovations of library science appear to have had little overlap with academic computing in the 1960s,[39] the notion of an information processing system for individuals grew more prevalent in the microcomputer hobbyist press of the mid-1970s. The hobbyist literature accompanied a growing computing counter-culture from which a database populism emerged that departed significantly from the institutional computing paradigm of the IBM/Hollerith tradition. Early hobbyist magazines such as *Byte*, *Dr. Dobb's Journal of Computer Calisthenics and Orthodontia*, and *Creative Computing* published a mix of practical and speculative material for the home computer enthusiast. Practical articles provided hands-on advice regarding the details of assembling a microcomputer kit while the speculative features presented visions of a computer-driven future waiting just on the other side of the soldering iron. For the hobbyist reader, these speculative pieces might have been a source of inspiration to motivate them through the frustrating process of building a home

[36] Neufeld and Cornog, "Database History," 183-190.
[37] H. G. Wells, "World Brain: The Idea of a Permanent World Encyclopaedia," *World Brain* (Garden City, N.Y: Doubleday, Doran & Co., 1938).
[38] Michael Buckland and Ziming Liu, "History of Information Science," in *Historical Studies in Information Science*, eds. Trudi Bellardo Hahn and Michael Buckland (The American Society for Information Science by Information Today, Inc., 1998): 272-295. See also the collection of materials on Buckland's website: http://people.ischool.berkeley.edu/~buckland/history.html.
[39] Douglas Engelbart's Augmentation Research Center at Stanford being a notable exception.

computer. "If you can just get the darn thing to work," the magazines seemed to say, "look at what you'll be able to do."

Ted M. Lau's "Total Kitchen Information System"[40] is typical of the speculative non-fiction pieces published in the first volume of *Byte* magazine in 1975. In a moment when databases remained exclusively the provenance of very large institutions, Lau's article presented a vision of the database turned toward popular ends. Describing the impetus for the article, Lau wrote,

> This project began as a gripe list my wife and I compiled after many frustrating experiences in the kitchen; throwing out spoiled food we'd forgotten in the refrigerator, abandoning a recipe for lack of a key ingredient, reeling with confusion after reading pages of grocery specials, neither being able to remember an appealing recipe nor to find it among our cookbooks, and so on.[41]

Lau goes on to describe, in detail, a computer system for processing all of the information related to the consumption of food in his home. Although he does not use the term, Lau's "TKIS" is essentially a small-scale database management system for the home. Admitting that the system exceeds the capacity of available microcomputers, Lau permits himself further speculation about how multiple small-scale database systems might be made interoperable throughout his community, "Instead of a paper receipt, the bag person at the market will plop a cassette in your bag containing all the items you purchased and their prices."[42]

Lau's description of the Total Kitchen Information System includes several key characteristics that distinguish personal databases from mass-scale information processing systems. Crucially, the TKIS is accessible to the non-specialist user. It is tailored towards the needs of individuals, is responsive to ad hoc queries (e.g., "Do I have all the ingredients I need to make quiche?"), and intended to reduce human labor and frustration. The infrastructure Lau imagines for the TKIS is considerably more modest than would be found in a contemporary institutional data processing facility. A microcomputer with a keyboard and video display would provide an interface into a collection of data stored on cassette tapes. Furthermore, Lau imagines that this commodity hardware will enable data to flow across contexts and institutional domains. The same cassettes used to store information at home are carried to the grocery store where they are helpfully updated based on the outcome of the shopping trip. Although such a comprehensive home data management system is still uncommon today, Lau's

[40] Ted M. Lau, "Total Kitchen Information System," in *The Best of Byte: Volume 1*, ed. David H. Ahl and Carl T. Helmers, Jr (Creative Computing Press, 1977), 360-363.
[41] Ibid., 360.
[42] Ibid., 363.

article provides a glimpse into the line of speculative thought that would later drive the popularization of database technologies in the 1980s.[43]

Another type of speculative feature directly engaged the role of the database in society and addressed the hobbyist reader as uniquely able to wrangle with the challenges that widespread computing might pose to individual liberty. Under David H. Ahl's editorial direction, *Creative Computing* magazine became the leading venue for this type of critical speculation, including a special issue dedicated to the "Computer Threat to Society" in 1975.[44] One exemplary article, titled "Surveys, The Census, and Privacy,"[45] consists of a lesson plan for a novice computer class. Ahl first introduces the general purpose of the census and, through five guided exercises, invites students to produce a simple census tabulation program on their own microcomputers. Along the way, Ahl provokes the student-reader to push past the practical challenge at hand to the broader implications of a national census,

> Would you want other members of the class to see your responses?... What should be done with [them]? Would you feel better if [they were] kept locked up by the teacher? Or would you rather see [them] burned? If [they were] destroyed and we decided later that we'd like to cross tabulate some results or add the results of two or three classes together to get a better overall average we wouldn't be able to. Then what?[46]

Ahl's constructivist approach is driven by the same assumption as more recent code literacy projects like Code Year: hands-on experience with the process of building software will enable people to think more critically about the role of software in society. The census project is notable, in particular, for its focus on data structures over code. Each of Ahl's hypothetical questions concerns the management of survey data rather than the algorithms used to analyze them.

---

[43] Customer loyalty systems deployed by U.S. retail chains offer a compelling contrast to the database populism of the TKIS. Whereas users of the TKIS manage their own data, customer loyalty systems centralize traces of customer behavior within an institutional database. Customers are afforded only very limited access to this information, usually in the form of promotional offers and other incentives. Customer loyalty programs became a locus of popular anxiety in early 2012 when the *New York Times* reported on efforts by the retail chain Target to predict when its customers were about to become pregnant. Charles Duhigg,"How Companies Learn Your Secrets," *New York Times*, February 16, 2012, accessed July 28, 2012, http://www.nytimes.com/2012/02/19/magazine/shopping-habits.html.

[44] See: *Creative Computing* 1, 6, (1975).

[45] David H. Ahl, "Surveys, The Census, and Privacy," in *The Best of Creative Computing, Volume 1*, ed. David H. Ahl, (Morristown, NJ: Creative Computing Press, 1976): 205-207.

[46] Ibid., 206.

The library science and hobbyist computing traditions each contributed alternative visions of the database in society. Rather than restrict access and centralize information, library scientists imagined and devised database technologies that would make institutional holdings more widely accessible to non-specialists. Readers of the hobbyist press, meanwhile, were presented with pragmatic visions of small, personal databases alongside commentary about the industrial and governmental use of mass-scale database technologies for social control. Often appearing in the same publications, this combination of the pragmatic and the political provided readers with a larger social context within which to understand and pursue their passion for technical mastery.

Databases for the non-specialist: SQL and the relational model

While hobbyist discourse produced a context for personal database management systems, the specific technical innovations needed to realize them came from a more unlikely source. IBM, descended as it was from Hollerith's Tabulating Machine Company, dedicated significant resources to the development and design of new data storage and information processing technologies. Beginning with Edgar F. Codd's 1970 paper on the relational model, a small team of researchers at IBM's San Jose research facility began to develop a collection of database technologies dubbed System R that they hoped would open the field of database management to non-specialist users.[47] Databases designed with Codd's relational model abstracted the technical details of physical storage from the logical relationships among the data being stored. This separation meant that everyday users of the system could manipulate information in the database without knowing anything about how it was actually being stored in the computer's memory, a significant departure from earlier systems.

Prevailing database management systems required some facility with a programming language like COBOL in order to make queries about the information they contained.[48] Furthermore, computerization restricted the punched card systems' capacity for ad-hoc querying. Most information processing

[47] During a reunion of the System R team in 1995, Mike Blasgen remarked, "Database wasn't a field yet in 1968, people at Berkeley called it 'data management' or 'file systems' or 'data banks'; the thing we call a 'database' really emerges from this milieu in the 1970s." See: Paul McJones, ed., "The 1995 SQL Reunion: People, Projects, and Politics," SRC Technical Note, 1997-018 (Digital Systems Research Center, August 20, 1997), accessed June 12, 2012, http://www.mcjones.org/System_R/SQL_Reunion_95/sqlr95.html.

[48] For a more comprehensive look at query languages in this early context, see: Ann S. Michaels, Benjamin Mittman, and C. Robert Carlson, "A Comparison of the Relational and CODASYL Approaches to Data-Base Management," *ACM Computer Surveys*, 8, 1 (1976): 125-151, doi:10.1145/356662.356668.

systems were "offline" which meant that most users had no direct access to the system. They would prepare a set of queries away from the computer, submit them to an "intermediary," and wait days (or weeks) for the results.[49] In the 1960s, a handful of systems came "online," which enabled users to iteratively refine their queries through interactive experimentation, but access to these systems was extremely limited and each one implemented its own idiosyncratic query method.[50]

The System R team hoped to address these constraints with SQL, a more accessible language for expressing queries to an online database. During a reunion of the System R team in 1995, Don Chamberlin recalled the database populism that drove the early stages of the project,

> What we thought we were doing [when designing SQL] was making it possible for non-programmers to interact with databases. We thought that this was going to open up access to data to a whole new class of people who could do things that were never possible before because they didn't know how to program.[51]

SQL syntax and grammar were refined through an extensive set of human-factors tests administered by linguist Phyllis Reisner[52] and the resulting language, initially called "SEQUEL" or "Structured English Query Language," was syntactically similar to imperative statements expressed in English. For example, given a simple database with information about flights in North America, a typical SQL query might read:

```
SELECT flightnumber
FROM flights
WHERE origin = 'BOS'
AND destination = 'LAX'
```

This simple example highlights the emphasis that the System R team placed on accessible interface design. Without knowing anything about the underlying database, a novice could make a reasonable guess as to the purpose of this query and begin to imagine queries of her own.[53]

---

[49] Griffiths and King, "US Information Retrieval System Evolution," 42.
[50] Ibid., 45-46.
[51] McJones, "1995 SQL Reunion."
[52] Phyllis Reisner, Raymond F. Boyce, and Donald D. Chamberlin, "Human Factors Evaluation of Two Data Base Query Languages—SQUARE and SEQUEL," *Proceedings of the AFIPS National Computer Conference, Anaheim, CA* (May 1975): 447.
[53] My choice of female pronoun here is intentional. Although computer programming was increasingly positioned as men's work, women were still frequently depicted as the operators of

By the mid-1970s, IBM's System R team had worked out the key pieces of SQL and the relational database model but it would be nearly another decade before IBM brought a product to market with these features. In the meantime, the team members published several academic papers and presented their work at conferences which, in turn, enabled smaller startup companies like Oracle to commercialize their ideas.[54] As a result, the first widely-available database management systems to use SQL-like syntax were not built for the industrial-scale mainframes like the ones that IBM sold to the U.S. government, but for mini- and micro-computers that were increasingly found in university labs, small businesses, and even the occasional kitchen table. As Blasgen recollected in 1995,

> The little Oracle thing ran on a machine that was the size of a carton of cigarettes. I remember because it was right there, stuck sideways onto the shelf. It was up on a little shelf above the desk, attached to a glass teletype. And that was all that it needed, and it ran fast, and I thought, "Simple, fast, cheap; that's neat. People will buy it." Exactly for the application that Roger mentioned: the query application.[55]

Blasgen is probably referring to a late-1970s model of the DEC PDP-11, which would have seemed "little" and "cheap" in contrast to the much larger mainframe computers being sold by IBM but was still not affordable enough for home use. Contemporary microcomputers still lacked the processing power and storage capacity required by relational database applications but Blasgen's emphasis on "the query application" is insightful. SQL-like syntax—often drawing only indirectly on the work of the System R team—proliferated among early personal database systems and became a common language for a new generation of information professionals.

---

computer terminals in promotional materials of the period. See: Thomas Haigh, "Masculinity and the Machine Man: Gender in the History of Data Processing," in *Gender Codes: Why Women Are Leaving Computing*, ed. Thomas J. Misa (Los Alamitos, CA: Wiley-IEEE Computer Society Press, 2010): 51-72.

[54] Some disagreement persists among the System R team as to the wisdom of this transparency from a business perspective but Mike Blasgen and Bob Yost argue convincingly that the unencumbered circulation of SQL in the 1970s created a market for databases that did not previously exist. McJones, "1995 SQL Reunion."

[55] Ibid.

Database populism and the brief dominance of the personal database

During the popularization of personal computing in the 1980s, databases were imagined in one of two forms: remote collections of information accessed via a telecommunication system,[56] or locally-produced, grassroots systems for the storage, analysis, and production of information. The latter form, which I am calling the "personal database," tended to run directly on a single-user home computer with no hard disk, while the former might run on a multi-user minicomputer like the PDP-11 with mass storage and a bank of modems. These two types of databases, accessible to non-specialists and built using commodity infrastructure, represented the joint realization of both the System R team's populist ambition and the hopeful speculation of the 1970s hobbyist press.

The diffusion of modem-equipped computers in offices, libraries, schools, and homes created new markets for commercially-produced databases. Subscription-based online services such as CompuServe and The Source served as "supermarkets" of data, providing access to job openings, real estate listings, newspaper archives, legal documents, movie reviews, and recipes.[57] For publishers, databases were as often sites of production as products. Editors working at terminals entered information directly into databases which, in turn, provided spell-checking and other word-processing functions.[58]

For many of the new non-specialist users, prevailing query methods such as SQL were still too confusing without the assistance of professional database intermediaries. Research into "natural language" query interfaces had been underway continuously since the 1960s but natural language search was still not widely implemented. Instead, a rising number of intermediaries were positioned on the boundary between databases and non-specialist users. Online services, such as CompuServe, produced software intermediaries in the form of constrained,

[56] One exception to the characterization of the database as a remote collection is found in editorials that anticipate the diffusion of CD-ROMs. For example, Mike Cane concludes the 1986 edition of the second volume of *The Computer Phone Book* with a short essay that predicts "any online system that is principally a remote depository for static information will, in a few years, discover its customer base evaporating... CD-ROM will do away with that by offering more information for a small fraction of the cost." Although Cane's prediction did not ultimately come to pass, the interrelationship of CD-ROMs and online databases during the 1990s is a subject deserving further scholarly attention. Mike Cane, editor, *The Computer Phone Book: Directory of Online Systems* (New York: Plume, 1986): 656.

[57] Neufeld and Cornog, "Database History," 187.

[58] Ibid., 186.

menu-driven interfaces while institutions such as public libraries employed human intermediaries to manage complex queries.[59]

While the proliferation of remote databases enabled popular access to vast collections of information, it lacked the critical edge of the hobbyists' database populism. Professional intermediaries and "supermarket" services made data more accessible by obscuring the underlying database infrastructure from non-specialists. Personal database software, meanwhile, invited home and small business users to produce their own databases. It was from this hands-on experience with small-scale databases that proponents of the computing counter-culture hoped non-specialists might develop a critique of much larger information processing systems.

C. J. Date's 1983 paperback *Database: A Primer* provides a helpful snapshot of the personal database at the start of the home computer era. As the author of a number of technical books about database management systems, Date and his publisher, Addison-Wesley, were careful to put a welcoming face on the new book to attract non-specialist readers. In the Foreword, Date draws a comparison between the database and another everyday marvel: the automobile,

> Fundamentally, there is no more need to know how a computer functions internally in order to use it than there is to know how the internal combustion engine works in order to drive a car.[60]

Date goes on to emphasize the pragmatic, rather than the technical, features of using a database. Paperwork is "drudgery" but a database is "efficient" and "quick."[61] For the home computer user, small business owner, or employee in a larger corporation, he writes, the goal of learning about databases is not to become versed in the details of a highly-technical subject, but simply "to get the system to do something useful."[62] The emphasis on home and small business use is reflected in the examples and exercises used throughout the book: creating an electronic address book, organizing one's music library, keeping an insurance inventory, or building a reference tool for scientific tables.

In the Introduction, Date specifically addresses the fear and anxiety that database technologies inspire in some readers. "A secondary objective of this book," he writes, "is to remove the mystery from database systems."[63] Date avoids discussing the institutions and uses with which information processing

[59] Griffiths and King, "US Information Retrieval System Evolution," 45.
[60] C. J. Date, *Database: A Primer* (Reading, MA: Addison-Wesley Publishing Company, 1986): vii.
[61] Ibid., back cover.
[62] Ibid., ix.
[63] Ibid., 4.

systems were historically associated and instead attributes the "considerable mystique" surrounding databases to the complex language used by computer scientists.[64] Attempting to settle the issue, Date returns to an emphasis on the pragmatic. "The whole point of all that complexity," he assures the reader, "is to make life easy for the user."[65]

Although SQL was not officially standardized until 1986 and no commercially-available implementation of SQL yet existed for the microcomputer in 1983, the majority of the code snippets in Date's book are written in SQL. This editorial decision reflects the bottom-up adoption of SQL as a presumptive standard by the database field more than a decade earlier. Jim Gray from IBM called SQL an "intergalactic dataspeak" which enabled interoperability among tools and software at a time when most systems were wholly incompatible with one another.[66] In addition to its technical advantages, grassroots standardization could have significant economic benefits for uncredentialed database technicians. Whereas operators of earlier information processing systems could not change jobs without an extensive period of adjustment, anyone conversant in SQL could transfer her skills among many different employers and database management systems.

Database populism and the personal database briefly flourished in the 1980s and, for many, it was the driving force behind the expensive purchase of a first microcomputer.[67] During the 1990s, however, the design and management of personal databases seemed to recede from everyday computing and return once again to the domain of the specialist. This transition did not diminish the importance of databases in computing culture but rather reflected a change in how database management systems were imagined, engineered, and marketed. On one hand, relational databases were believed to be too complex for the needs of the everyday user. Instead, users were encouraged to store their personal data in spreadsheets or flat text files. On the other, remote databases tended to provide hierarchical menus and natural language search tools rather than command-line interfaces that could accept queries in SQL. As the number of database specialists—many of whom were likely self-trained hobbyists—grew in response to the demand for database intermediaries, providers of database technologies abandoned the popular accessibility that had marked the previous two decades of development.

---

[64] Ibid.
[65] Ibid., 4.
[66] McJones, "1995 SQL Reunion."
[67] "Database management is one of the most important functions provided by modern computer systems...very often it is the principal justification for acquiring the computer in the first place." Date, *Database*, 3.

Russ Walter's *Secret Guide to Computers* is a large, off-beat encyclopedia of personal computing that has been in continuous print for more than thirty-one editions.[68] The contrast in Walter's discussion of databases from the 1984 to the 1995 editions of his book offers some evidence that providers of database software favored the needs of institutional customers over individual users. In the 1984 edition, Walter introduces the database concept using the metaphor of the standard office filing cabinet and, like Date, emphasizes the practical advantages of maintaining a database for small business use.[69] He concludes with a short discussion of the relational data model and offers some advice for choosing from among the handful of available personal database products.[70] In the 1995 edition, coverage of databases ballooned from five to twenty-nine pages, including detailed discussion of a widely-available SQL-inspired query language.[71] In spite of the increased attention and technical detail, Walter discouraged his readers from purchasing the latest versions of popular database software because of a lack of popular accessibility,

> They're powerful, fancy, and more than most folks can understand. If you buy one of them, you'll probably admire the big box it comes in, put it on the shelf, and invite friends to visit you and admire your big box, but you'll never figure out how to use it.[72]

Although Walter's enthusiasm for small-scale databases did not flag between the two editions, he struggled through more than a dozen new database products before concluding that readers purchase Q&A, a database management system from the 1980s,

> Since computers were supposed to make our lives easy, I recommend you get one of the easy database programs[.] I run my entire business using Q&A for DOS and never found any need to switch.[73]

Two additional factors appear to have contributed to the gradual disappearance of the personal database and attendant reinscription of database management as a specialist activity at the end of the 1990s. First, spreadsheet programs such as

---

[68] The most recent edition of *The Secret Guide to Computers* was published in 2012. For more information on Walter's publication record, see: http://www.angelfire.com/nh/secret/.

[69] Russ Walter, *The Secret Guide to Computers, Volume 2: Deep Secrets, 11th edition* (Self-published, 1984): 62-63.

[70] Ibid., 63. Also like Date, Walter remarks only offhandedly about the privacy implications of mass-scale data processing.

[71] Russ Walter, *The Secret Guide to Computers, 20th edition* (Self-published, 1995): 206-219, 450-465, 552.

[72] Ibid., 219.

[73] Ibid.

Microsoft Excel increasingly incorporated data management features that were previously available only in database software. For small tasks, such as assembling a club roster, it became more convenient to create a spreadsheet than a relational database. Second, the turn toward web-based services produced a new discursive distinction between "front end" and "back end" software components. With the database clearly fixed in the "back end," there appeared to be little reason to develop tools for the non-specialist and database management systems continued to grow more arcane.

Although use of database technologies became widespread during the 1980s, there are fewer examples of publications like *Creative Computing* that paired pragmatic technical features with social critique. And while cyberpunk science fiction authors continued to associate mass-scale data processing with totalitarianism, most popular non-fiction tended, as in Date's book, to focus primarily on the pragmatic benefits that database technologies could offer to individuals. That the database populism of the 1970s hobbyist literature appeared most clearly in the marketplace of the 1980s—in the form of new products, services, and professional opportunities—does not, however, indicate that it was solely, or even predominantly, a commercial phenomenon. One unexpected aspect of the period was the growth of database intermediaries.[74] This group, inclusive of programmers, information scientists, and specialty librarians, was employed across public and private institutions to bridge the gap between databases and their users. Whereas universal data literacy of the sort imagined by some hobbyists in the 1970s may not have been realized, this new class of information professionals was tasked with providing personalized assistance and customized software interfaces to otherwise inaccessible data resources. Their role, in other words, was novel: to represent the interests of non-specialists within institutions producing and distributing large collections of information.

Abandoning SQL and the relational model: The return of mass-scale information processing?

At the 3rd international conference on Very Large Data Bases in 2007, Michael Stonebraker et al. presented a paper titled, "The End of an Architectural Era (It's Time for a Complete Rewrite)" in which they argued that the dominant relational database management systems were being stretched beyond their usefulness and ought to be retired in favor of custom database software narrowly-designed to the

[74] Neufeld and Cornog, "Database History," 186.

needs of specific institutions and uses.[75] On its face, the argument seems to call for a return to the early-20th century era of data processing, during which institutions built and maintained their own idiosyncratic tools,

> We conclude that the current [relational database management systems,] while attempting to be a 'one size fits all' solution, in fact, excel at nothing. Hence, they are 25 year old legacy code lines that should be retired in favor of a collection of "from scratch" specialized engines."[76]

But Stonebraker et al.'s presentation appeared during the start of a multi-year explosion of discourse and software among engineers—especially those employed in the production of intermediaries—regarding the inadequacy of existing databases for massive communication systems on the web. Unable to manage the volume and throughput of data being generated by their users, highly-centralized systems such as Google, Facebook, and Amazon were doing just what Stonebraker et al. suggested: producing custom databases that did away with SQL and the relational model. The new database management systems came to be known collectively as "NoSQL" systems.[77]

The various technologies associated with the NoSQL "movement"[78] share more with each other culturally than they do technically but it is important to note a few common features. NoSQL databases tend to be implemented in situations where billions of small chunks of information need to be inserted into or read from a database at a very high speed (e.g. millions of users clicking "Like" on Facebook every minute.) The typical hardware infrastructure for NoSQL systems is a cluster of hundreds (or thousands) of relatively low-cost PCs running free software rather than fewer, more high powered servers. In most scenarios, new machines can be added to, or subtracted from, the cluster without interrupting the overall system. NoSQL databases tend to forego the relational model and an accessible query language, the two key contributions of the System R team,

---

[75] Michael Stonebraker, Samuel Madden, Daniel J. Abadi, Stavros Harizopoulos, Nabil Hachem, and Pat Helland, "The end of an architectural era: (it's time for a complete rewrite)," in *Proceedings of the 33rd international conference on Very large data bases (VLDB '07)* (VLDB Endowment, 2007): 1150-1160.

[76] Ibid., 1150.

[77] For a thorough overview of the literature that informs the NoSQL approach, see the list of papers from the 2010 NoSQL summer reading club, accessed June 12, 2012, http://nosqlsummer.org/papers.

[78] "NoSQL" is a contentious term in some contexts. For an example of a positive use of the term "movement," see Alex Popescu, "NoSQL." *MyNoSQL*, accessed June 12, 2012, http://nosql.mypopescu.com/kb/nosql. For a derisive example, see: Carlo Strozzi, "Philosophy of NoSQL," accessed June 12, 2012, http://www.strozzi.it/cgi-bin/CSA/tw7/I/en_US/NoSQL/Philosophy%20of%20NoSQL.

arguing that they represent unnecessary computational overhead for data intensive applications. Instead of storing data according to a set of pre-determined relationships, many NoSQL systems simply store strings of bytes and assume that client software will be able to extract meaning from them.[79] In the place of a dedicated query language like SQL, most NoSQL databases are queried from within a general-purpose programming language.

While Stonebraker et al. made a rational case for abandoning the relational model, the advocacy of others—notably database professionals—was marked by a surprising amount of passion and anger. In thousands of blog posts, tweets, and videos, intermediary database programmers decried the "horrors" of working with existing relational database management systems such as the widely-deployed MySQL.[80] Such strangely forceful reactions brought a moral weight to the choice of database software indicating that more than mere technical preferences were at stake. Contrasting the populist vision of System R with the day-to-day experience of database programmers in the early 2000s helps illuminate the latent stakes of this moral conflict.

While the System R team created SQL as an accessible means for non-programmers to express queries to the database, few commercial database products used SQL as a primary interface, especially after the diffusion of graphical user interfaces in the late-1980s.[81] Rather than serve as an interactive language for everyday database use, SQL statements were often embedded within software written in a more general-purpose programming language such as C or Java. In practice, then, a lexical language designed to be an interface for non-programmers was being used almost exclusively by programmers in the course of building intermediary graphical interfaces. This confusing mismatch may be one explanation for the frustration felt by so many database professionals—SQL was not designed for specialist use and yet it was required by most major database

[79] A paper by Chang et al. describing Google's BigTable system provided an important early model of this non-relational approach. Fay Chang, Jeffrey Dean, Sanjay Ghemawat, Wilson C. Hsieh, Deborah A. Wallach, Mike Burrows, Tushar Chandra, Andrew Fikes, and Robert E. Gruber, "Bigtable: a distributed storage system for structured data," in *Proceedings of the 7th USENIX Symposium on Operating Systems Design and Implementation - Volume 7 (OSDI '06)* (Berkeley, CA: USENIX Association, 2006), 205-218, accessed August 22, 2012, http://usenix.org/event/osdi06/tech/chang/chang.pdf.

[80] In the full context of this quote, Williams refers to specific technical limitations of MySQL: "This [new project] has given us the luxury of building against a NOSQL database, which means we can put the horrors of MySQL sharding and expensive scalability behind us." Dominic Williams, "HBase vs Cassandra: why we moved," *Dominic Williams,* February 24, 2010, accessed June 12, 2012, http://ria101.wordpress.com/2010/02/24/hbase-vs-cassandra-why-we-moved/.

[81] During the 1995 reunion, Chamberlin remarked that the team did not anticipate the accessibility that a GUI might afford. McJones, "1995 SQL Reunion."

management systems. By dropping SQL and allowing programmers to express queries in their preferred programming languages, NoSQL systems actually provided a more accessible interface for their primary users—database programmers—than did the descendants of System R.

Although abandoning SQL and the relational model in favor of specialized systems might initially seem like a return to an earlier era of inaccessible databases and very expensive infrastructures, the enthusiastic reaction of so many intermediary programmers invites a closer examination of the social dimensions of this transition. In the course of a presentation from 2004 about the MapReduce programming model used in many of Google's data processing tools, Jay Dean and Sanjay Ghemawat offer some insight into the experience of working with a NoSQL system at the start of the 21st century.[82] MapReduce is the name of an approach to processing very large collections of data using several hundred (or thousand) computers in a cluster. For programmers, the crucial advantage of the MapReduce model is that an underlying system manages most of the complexity of mass-scale data processing automatically. Dean and Ghemawat describe the experience of writing MapReduce code as "fun" and emphasize that MapReduce makes mass-scale data processing "easy to understand."[83] The snippets of code they share are each only a few lines long and would be easily comprehendible to most programmers in the audience.

The focus on pleasure in Dean and Ghemawat's presentation suggests that the passion of the NoSQL movement may be driven more by the affective than the technical aspects of communicating with a database. Indeed, in a blog post from 2010, Stonebraker argued that abandoning SQL was not necessary to create a database capable of efficiently processing mass-scale "big data"[84] so any perceived technical limitations are insufficient to explain the widespread enthusiasm for its exclusion. It may be the case that intermediary programmers advocate for NoSQL systems principally because writing code in the style of MapReduce is more fun than composing queries in SQL.[85]

Given that SQL was not intended to be a tool for programmers, it may be best to understand the turn to NoSQL databases not as a rejection but a revision of

---

[82] Jay Dean and Sanjay Ghemawat, "MapReduce: Simplified Data Processing on Large Clusters," *OSDI'04: Sixth Symposium on Operating System Design and Implementation* (2004), accessed June 12, 2012, http://research.google.com/archive/mapreduce.html.

[83] Ibid.

[84] Michael Stonebraker, "SQL databases v. NoSQL databases," *Communications of the ACM*, 53, 4, (2010), 10-11, doi:10.1145/1721654.172144659.

[85] Many programmers report that "fun" is an important motivating force in their work. For a particularly compelling example, see Kelty's discussion of the role of fun in the Linux kernel project. Christopher M. Kelty, *Two Bits: The Cultural Significance of Free Software* (Durham, NC: Duke University Press, 2008).

System R's database populism. Rather than attempt to create a single universal interface for all users, NoSQL systems create a more generative intermediary layer in which programmers can produce multiple custom interfaces that, in turn, make the database more accessible to a greater diversity of users. Whether or not this potential plurality of interfaces is realized within the current industrial context remains to be seen.

The possibility of a revived database populism

At the heart of mass-scale data processing today is a fundamental conflict between complexity and simplicity. Whereas each individual computer in a MapReduce cluster is an affordable, off-the-shelf PC, building and maintaining thousands of such systems in a network is beyond the capacity of any individual or small business owner. Similarly, the infrastructure that enables search engines like Google and Bing to quickly interpret natural language queries requires persistent, highly-capitalized institutions staffed by relatively small numbers of specialists. And whereas the NoSQL approach to data management is intuitive and accessible to experienced programmers, it leaves non-specialists reliant on intermediary software interfaces. In sum, there is little overt consideration in the mass-scale database systems of the 2010s for the database populism of the 1970s and 1980s.

For the most part, the transition from relational, SQL-driven databases to custom-built, NoSQL systems has occurred outside the view of both scholars of communication and everyday users of large-scale communication systems. As a result, we know very little about what impact, if any, this transition is having on the social lives of the users and programmers of sites and services like Twitter, Google, Amazon, and Facebook. While a sub-field of communication research is emerging in response to the discourses of "big data,"[86] it has not yet begun attending to the infrastructures that undergird the aggregation and analysis of mass-scale data sets. Yet, as this history demonstrates, infrastructure provides a

[86] Two useful entry-points into this field: (a) the work of the Software Studies Initiative to produce "cultural analytics" methodologies for studying big cultural data; and (b) boyd and Crawford's cautionary response to the over-enthusiastic embrace of big data. Lev Manovich, "How to Follow Software Users? (Digital Humanities, Software Studies, Big Data)," unpublished paper, Software Studies Initiative, 2012, accessed June 12, 2012, http://www.softwarestudies.com/cultural_analytics/Manovich.How_to_Follow_Software_Users; danah boyd and Kate Crawford, "Six Provocations for Big Data," *A Decade in Internet Time: Symposium on the Dynamics of the Internet and Society*, September, 2011, accessed June 12, 2012, doi:10.2139/ssrn.1926431.

powerful analytic lens through which the discourses of "big data" are tempered by more than a century of databases in society.[87]

Fear remains a crucial vector for understanding mass-scale information processing. As in the anti-war period, loss of privacy and self-determination continue to be sources of anxiety and today they are paired with contradictory messages regarding the social benefits of massive data processing by state and private actors. In a puzzling press release announcing new privacy features, Facebook included a quote from CEO Mark Zuckerberg that encouraged people to continue giving up their privacy, "When people share more," he wrote, "the world becomes more open and connected."[88] The abstract value of "openness" is also found in military-sponsored research that promises increased safety and security. Early in 2012, the Department of Defense announced that it is placing a "big bet on big data" as a means to anticipate and identify "cyber threats" through massive processing of online communication data.[89] While the defensive power of such a system has not been demonstrated, the threat it poses to individual liberty is evidenced by projects such as Jernigan and Mistree's method for predicting sexual orientation using publicly-available Facebook data.[90]

In spite of the very reasonable fear historically associated with mass-scale information processing, we should not expect users to retreat from such systems—quite the contrary.[91] From the perspective of information retrieval,

---

[87] In spite of its name and reputation, systems associated with "big data" tend to have very short memories. Twitter is scarcely six years old but its infrastructure is incapable of providing users with their complete history on demand. To access one's full Twitter history, a fax must be sent to the Twitter Legal department with a valid photo ID. For more detail on this process and its outcomes, see: Anne Helmond, "What does Twitter know about me? My .zip file with 50Mb of data," *New Media Research blog*, April 17, 2012, accessed June 12, 2012, http://www.annehelmond.nl/2012/04/17/what-does-twitter-know-about-me-my-zip-file-with-50mb-of-data/.

[88] Facebook, "Facebook Redesigns Privacy," *Facebook Newsroom*, May 26, 2010, accessed June 12, 2012, http://newsroom.fb.com/News/Facebook-Redesigns-Privacy-bb.aspx.

[89] "Fact Sheet: Big Data Across the Federal Government," *Office of Science and Technology Policy, Executive Office of the President*, March 29, 2012, accessed June 12, 2012, http://www.whitehouse.gov/sites/default/files/microsites/ostp/big_data_fact_sheet_final.pdf.

[90] Carter Jernigan and Behram F. T. Mistree, "Gaydar: Facebook friendships expose sexual orientation," *First Monday* 14, 10 (2009).

[91] For evidence of the ambivalence that some users feel about data-mining by institutions like Facebook, see the provocative reader comments attached to coverage of the Jernigan and Mistree's project on *The Telegraph* and *Boston Globe:* Carolyn Y. Johnson, "Project 'Gaydar'," *The Boston Globe*, September 20, 2009, accessed on June 12, 2012, http://www.boston.com/bostonglobe/ideas/articles/2009/09/20/project_gaydar_an_mit_experiment_raises_new_questions_about_online_privacy/; Matthew Moore, "Gay men 'can be identified by their Facebook friends'," *The Telegraph*, September 21, 2009, accessed on June

Google's natural language search tools may be the most widely-accessible "supermarket" data service ever created. For many users, even if the full extent of its data-mining efforts were known, the value of Google search might still outweigh the attendant loss of privacy. Unlike users of 1980s "supermarket" data services like CompuServe, however, users of today's mass-scale search engines are not also engaged with the production of their own personal databases. This lack of hands-on database experience makes it difficult for users to critically evaluate the kinds of mass-scale information processing practiced by Google and others.

In its various manifestations, database populism always involves consideration for both pragmatic and critical concerns. On one hand, critics such as Pariser and Kovacs advocate for a defensive stance against invasive data processing and urge users to modify their behavior in order to avoid observation. On the other, pragmatists such as Date and Walter focus on the benefits that database technologies offer to individual users with little consideration for their social impact. The database populism envisioned in the hobbyist literature of the 1970s, however, represents a balanced view. For participants in the microcomputing counter-culture, hands-on experience with small databases provided a practical foundation for an informed critique of the mass-scale data processing systems in society.

Databases represent part of a largely transparent infrastructure on which a considerable volume of social activity depends. As Date remarked in 1983, the database "is very much a human problem, not a system problem."[92] Renewed database populism for the 2010s, then, must begin with the lived experiences of database users and programmers, not a critique of database technologies.

12, 2012, http://www.telegraph.co.uk/technology/facebook/6213590/Gay-men-can-be-identified-by-their-Facebook-friends.html.

[92] Date, *Database*, 217.

Bibliography

Ahl, David H. “Surveys, The Census, and Privacy.” In *The Best of Creative Computing, Volume 1*, edited by David H. Ahl, 205-207. Morristown, NJ: Creative Computing Press, 1976.

Atwood, Jeff. “Please Don't Learn to Code.” *Coding Horror*, May 15, 2012. Accessed June 12, 2012. http://www.codinghorror.com/blog/2012/05/please-dont-learn-to-code.html.

Beniger, James R. *The Control Revolution*. Cambridge: Harvard University Press, 1986.

boyd, danah and Kate Crawford. “Six Provocations for Big Data.” *A Decade in Internet Time: Symposium on the Dynamics of the Internet and Society*, September, 2011. doi:10.2139/ssrn.1926431.

Popova, Maria. “The Filter Bubble: Algorithm vs. Curator & the Value of Serendipity.” *Brain Pickings* (blog), May 12, 2011. Accessed July 26, 2012. http://www.brainpickings.org/index.php/2011/05/12/the-filter-bubble/

Buckland, Michael and Ziming Liu. “History of Information Science.” In *Historical Studies in Information Science*, edited by Trudi Bellardo Hahn and Michael Buckland, 272-295. The American Society for Information Science by Information Today, Inc., 1998.

Cane, Mike, ed. *The Computer Phone Book: Directory of Online Systems.* New York: Plume, 1986.

Chang, Fay, Jeffrey Dean, Sanjay Ghemawat, Wilson C. Hsieh, Deborah A. Wallach, Mike Burrows, Tushar Chandra, Andrew Fikes, and Robert E. Gruber. “Bigtable: a distributed storage system for structured data.” In *Proceedings of the 7th USENIX Symposium on Operating Systems Design and Implementation - Volume 7 (OSDI '06)*, 205-218. Berkeley, CA: USENIX Association, 2006. Accessed August 22, 2012. http://usenix.org/event/osdi06/tech/chang/chang.pdf.

Cheney-Lippold, John. “A New Algorithmic Identity Soft Biopolitics and the Modulation of Control.” *Theory, Culture & Society*, 28, 6 (2011): 164-181. doi:10.1177/0263276411424420.

Codd, E. F. “A Relational Model of Data for Large Shared Data Banks.” *Communications of ACM*, 13, 6 (1970): 377-387.

Date, C. J. *Database: A Primer*. Reading, MA: Addison-Wesley Publishing Company, 1986.

DBLP Computer Science Bibliography, The. "ACM SIGMOD Conference." Accessed June 12, 2012. http://www.informatik.uni-trier.de/~ley/db/conf/sigmod/.

Dean, Jay and Sanjay Ghemawat. "MapReduce: Simplified Data Processing on Large Clusters," *OSDI'04: Sixth Symposium on Operating System Design and Implementation*, San Francisco, CA, December, 2004. Accessed July 28, 2012. http://research.google.com/archive/mapreduce.html.

Dorman, Robert L. "The Creation and Destruction of the 1890 Federal Census." *The American Archivist* 71, 2 (2008): 350-383.

Duhigg, Charles. "How Companies Learn Your Secrets." *New York Times*, February 16, 2012. Accessed July 28, 2012. http://www.nytimes.com/2012/02/19/magazine/shopping-habits.html.

Elective-C, May 15, 2012 (3:04 a.m.), comment on Jeff Atwood, "Please Don't Learn to Code," *Coding Horror*, May 15, 2012, http://www.codinghorror.com/blog/2012/05/please-dont-learn-to-code.html.

Facebook. "Facebook Redesigns Privacy." *Facebook Newsroom,* May 26, 2010. Accessed June 12, 2012. http://newsroom.fb.com/News/Facebook-Redesigns-Privacy-bb.aspx.

"Fact Sheet: Big Data Across the Federal Government." *Office of Science and Technology Policy, Executive Office of the President,* March 29, 2012. Accessed June 12, 2012. http://www.whitehouse.gov/sites/default/files/microsites/ostp/big_data_fact_sheet_final.pdf.

Gillespie, Tarleton. "Can an Algorithm be Wrong?" *Limn* 2, 2012. Accessed June 12, 2012. http://limn.it/can-an-algorithm-be-wrong/.

Griffiths, José-Marie, and Donald W. King. "US Information Retrieval System Evolution and Evaluation (1945-1975)." *IEEE Annals of the History of Computing* 24, 3 (2002): 35-55.

Haigh, Thomas. "Masculinity and the Machine Man: Gender in the History of Data Processing." In *Gender Codes: Why Women Are Leaving Computing*, edited by Thomas J. Misa, 51-72. Los Alamitos, CA: Wiley-IEEE Computer Society Press, 2010.

Helmond, Anne. “What does Twitter know about me? My .zip file with 50Mb of data.” *New Media Research blog*, April 17, 2012. Accessed June 12, 2012. http://www.annehelmond.nl/2012/04/17/what-does-twitter-know-about-me-my-zip-file-with-50mb-of-data/.

Jernigan, Carter and Behram F. T. Mistree. “Gaydar: Facebook friendships expose sexual orientation.” *First Monday* 14, 10 (2009).

Johnson, Carolyn Y. “Project ‘Gaydar’.” *The Boston Globe*, September 20, 2009. Accessed June 12, 2012.

http://www.boston.com/bostonglobe/ideas/articles/2009/09/20/project_gaydar_an_mit_experiment_raises_new_questions_about_online_privacy/.

Kelty, Christopher M. *Two Bits: The Cultural Significance of Free Software.* Durham, NC: Duke University Press, 2008.

Knuth, Donald. *The Art of Computer Programming, Volume 1: Fundamental Algorithms*. Reading, MA: Addison-Wesley, 1973.

Koch, Tom and Bob Clark. “We’re Losing Our War Against Computers.” In *The Best of Creative Computing: Volume 1*, edited by David H. Ahl, 122-123. Morristown, NJ: Creative Computing Press, 1976.

Kovacs, Gary. “Gary Kovacs: Tracking the Trackers.” *TED*. Accessed June 12, 2012. http://www.ted.com/talks/gary_kovacs_tracking_the_trackers.html.

Lau, Ted M. “Total Kitchen Information System.” In *The Best of Byte: Volume 1*, edited by David H. Ahl and Carl T. Helmers, Jr, 360-363. Morristown, NJ: Creative Computing Press, 1977.

Lotan, Gilad. “Data Reveals That ‘Occupying’ Twitter Trending Topics is Harder Than It Looks!” *SocialFlow* (blog), October 12, 2011. Accessed June 12, 2012. http://blog.socialflow.com/post/7120244374/data-reeals-that-occupying-twitter-trending-topics-is-harder-than-it-looks.

Manovich, Lev. “Database as Symbolic Form.” *Convergence* 5, 2 (1999): 89-99. doi:10.1177/135485659900500206.

—. “How to Follow Software Users? (Digital Humanities, Software Studies, Big Data).” Unpublished paper, Software Studies Initiative, 2012. Accessed June 12, 2012. http://www.softwarestudies.com/cultural_analytics/Manovich.How_to_Follow_Software_Users.doc.

Markoff, John. *What the Dormouse Said: How the 60s Counterculture Shaped the Personal Computer Industry*. New York: Viking, 2005.

McJones, Paul, ed. "The 1995 SQL Reunion: People, Projects, and Politics." SRC Technical Note, 1997-018 (Digital Systems Research Center, August 20, 1997). Accessed June 12, 2012. http://www.mcjones.org/System_R/SQL_Reunion_95/sqlr95.html.

Michaels, Ann S., Benjamin Mittman, and C. Robert Carlson. "A Comparison of the Relational and CODASYL Approaches to Data-Base Management." *ACM Computer Surveys* 8, 1 (1976): 125-151. doi:10.1145/356662.356668.

Moore, Matthew. "Gay men 'can be identified by their Facebook friends'." *The Telegraph*, September 21, 2009. Accessed June 12, 2012. http://www.telegraph.co.uk/technology/facebook/6213590/Gay-men-can-be-identified-by-their-Facebook-friends.html.

Nelson, Theodor H.. *Computer Lib.* Self-published, 1974.

Neufeld, M. Lynne, and Martha Cornog. "Database history: From dinosaurs to compact discs." *Journal of the American Society for Information Science* 37, 4 (1986): 183-190. doi: 10.1002/(SICI)1097-4571(198607)37:4<183::AID-ASI2>3.0.CO;2-W.

Pariser, Eli. *The Filter Bubble: What the Internet Is Hiding from You*. New York: Penguin, 2011.

—. "10 Ways to Pop Your Filter Bubble." *The Filter Bubble*. Accessed June 12, 2012. http://www.thefilterbubble.com/10-things-you-can-do.

Raymond, Eric S., ed. "GIGO." *The Jargon File.* Accessed July 21, 2012. http://catb.org/jargon/html/G/GIGO.html.

Reisner, P., R. F. Boyce, and D. D. Chamberlin. "Human Factors Evaluation of Two Data Base Query Languages—SQUARE and SEQUEL." *Proceedings of the AFIPS National Computer Conference*, *Anaheim, CA* (May 1975): 447.

Rushkoff, Douglas. *Program or Be Programmed: Ten Commands for a Digital Age*. Berkeley, CA: Soft Skull Press, 2011.

Segall, Laurie. "Code Year Draws 200,000 aspiring programmers." *CNN Money,* January 6, 2012. Accessed June 12, 2012. http://money.cnn.com/2012/01/06/technology/code_year/index.htm.

Star, Susan Leigh. "The ethnography of infrastructure." *American Behavioral Scientist,* 43, 3 (1999): 377. doi:10.1177/00027649921955326.

Stonebraker, Michael, Samuel Madden, Daniel J. Abadi, Stavros Harizopoulos, Nabil Hachem, and Pat Helland. "The end of an architectural era: (it's time

for a complete rewrite).” In *Proceedings of the 33rd international conference on Very large data bases (VLDB '07),* 1150-1160. VLDB Endowment, 2007.

—. “SQL databases v. NoSQL databases.” *Communications of the ACM* 53, 4, (2010): 10-11. doi:10.1145/1721654.1721659.

Streeter, Thomas. *The Net Effect: Romanticism, Capitalism, and the Internet.* New York: New York University Press, 2011.

Strozzi, Carlo. “NoSQL: A non-SQL RDBMS.” Accessed June 12, 2012. http://www.strozzi.it/cgi-bin/CSA/tw7/I/en_US/NoSQL/.

Turner, Fred. *From Counterculture to Cyberculture: Stewart Brand, the Whole Earth Network, and the Rise of Digital Utopianism*. Chicago: University of Chicago Press, 2006.

Twitter. “To Trend or Not to Trend….” *Twitter* (blog), December 8, 2010. Accessed July 26, 2012. http://blog.twitter.com/2010/12/to-trend-or-not-to-trend.html.

Waldrop, M. Mitchell. *The Dream Machine: J. C. R. Licklider and the Revolution That Made Computing Personal*. New York: Penguin, 2001.

Walter, Russ. *The Secret Guide to Computers, Volume 2: Deep Secrets, 11th edition*. Self-published, 1984.

—. *The Secret Guide to Computers, 20th edition*. Self-published, 1995.

Wells, H. G.. “World Brain: The Idea of a Permanent World Encyclopaedia.” *World Brain*. Garden City, N.Y: Doubleday, Doran & Co., 1938.

Williams, Dominic. “HBase vs Cassandra: why we moved.” February 24, 2010. Accessed June 12, 2012. http://ria101.wordpress.com/2010/02/24/hbase-vs-cassandra-why-we-moved/.