DOI:10.1145/1516046.1516062

RAKESH AGRAWAL
ANASTASIA AILAMAKI
PHILIP A. BERNSTEIN
ERIC A. BREWER
MICHAEL J. CAREY
SURAJIT CHAUDHURI
ANHAI DOAN
DANIELA FLORESCU
MICHAEL J. FRANKLIN
HECTOR GARCIA-MOLINA
JOHANNES GEHRKE
LE GRUENWALD
LAURA M. HAAS
ALON Y. HALEVY
JOSEPH M. HELLERSTEIN
YANNIS E. IOANNIDIS
HANK F. KORTH
DONALD KOSSMANN
SAMUEL MADDEN
ROGER MAGOULAS
BENG CHIN OOI
TIM O'REILLY
RAGHU RAMAKRISHNAN
SUNITA SARAWAGI
MICHAEL STONEBRAKER
ALEXANDER S. SZALAY
GERHARD WEIKUM

Database research is expanding, with major efforts in system architecture, new languages, cloud services, mobile and virtual worlds, and interplay between structure and text.

The Claremont Report on Database Research

A GROUP OF database researchers, architects, users, and pundits met in May 2008 at the Claremont Resort in Berkeley, CA, to discuss the state of database research and its effects on practice. This was the seventh meeting of this sort over the past 20 years and was distinguished by a broad consensus that the database community is at a turning point in its history, due toboth an explosion of data and usage scenarios and major shifts in computing hardware and platforms.

Here, we explore the conclusions of this self-assessment. It is by definition somewhat inward-focused but may be of interest to the broader computing community as both a window into upcoming directions in database research and

a description of some of the community issues and initiatives that surfaced. We describe the group's consensus view of new focus areas for research, including database engine architectures, declarative programming languages, interplay of structured data and free text, cloud data services, and mobile and virtual worlds. We also report on discussions of the database community's growth and processes that may be of interest to other research areas facing similar challenges.

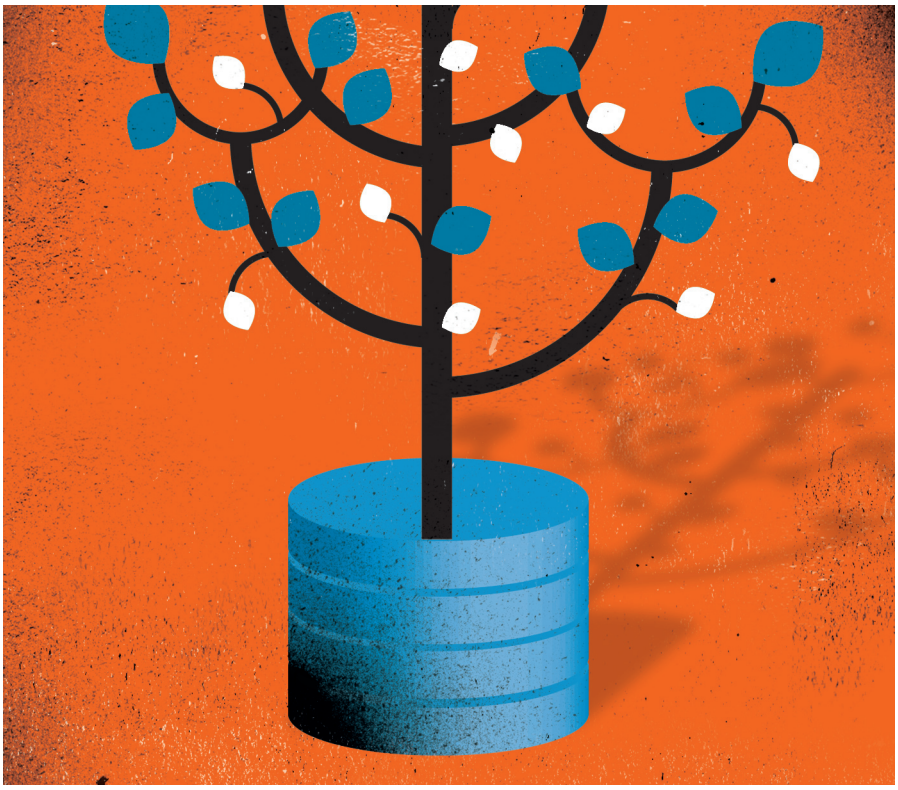
Over the past 20 years, small groups of database researchers have periodically gathered to assess the state of the field and propose directions for future research.[1,3–7] Reports of the meetings served to foster debate within the database research community, explain research directions to external organizations, and help focus community efforts on timely challenges.

The theme of the Claremont meeting was that database research and the data-management industry are at a turning point, with unusually rich opportunities for technical advances, intellectual achievement, entrepreneurship, and benefits for science and society. Given the large number of opportunities, it is important for the database research community to address issues that maximize relevance within the field, across computing, and in external fields as well.

The sense of change that emerged in the meeting was a function of several factors:

Excitement over "big data." In recent years, the number of communities working with large volumes of data has grown considerably to include not only traditional enterprise applications and Web search but also e-science efforts (in astronomy, biology, earth science, and more), digital entertainment, natural-language processing, and social-network analysis. While the user base for traditional database management systems (DBMSs) is growing quickly, there is also a groundswell of effort to design new custom data-management solutions from simpler components. The ubiquity of big data is expanding the base of users and developers of data-management technologies and will undoubtedly shake up the database research field.

Data analysis as profit center. In traditional enterprise settings, the barriers between IT departments and business units are coming down, and there are many examples of companies where data is indeed the business itself. As a consequence, data capture, integration, and analysis are no longer viewed as a business cost but as the keys to efficiency and profit. The value of software to support data analytics has been growing as a result. In 2007, corporate acquisitions of business-intelligence vendors alone totaled $15 billion,[2] and



that is only the "front end" of the data-analytics tool chain. Market pressure for better analytics also brings new users to the technology with new demands. Statistically sophisticated analysts are being hired in a growing number of industries, with increasing interest in running their formulae on the raw data. At the same time, a growing number of nontechnical decision makers want to "get their hands on the numbers" as well in simple and intuitive ways.

Ubiquity of structured and unstructured data. There is an explosion of structured data on the Web and on enterprise intranets. This data is from a variety of sources beyond traditional databases, including large-scale efforts to extract structured information from text, software logs and sensors, and crawls of deep-Web sites. There is also an explosion of text-focused semistructured data in the public domain in the form of blogs, Web 2.0 communities, and instant messaging. New incentive structures and Web sites have emerged for publishing and curating structured data in a shared fashion as well. Text-centric approaches to managing the data are easy to use but ignore latent structure in the data that might add significant value. The race is on to develop techniques that extract useful data from mostly noisy text and structured corpora, enable deeper exploration into individual data sets, and connect data sets together to wring out as much value as possible.

Expanded developer demands. Programmer adoption of relational DBMSs and query languages has grown significantly in recent years, accelerated by the maturation of open source systems (such as MySQL and PostgreSQL) and the growing popularity of object-relational mapping packages (such as Ruby on Rails). However, the expanded user base brings new expectations for programmability and usability from a larger, broader, less-specialized community of programmers.

Some of them are unhappy or unwilling to "drop into" SQL, viewing DBMSs

ILLUSTRATION BY GLUEKIT

as unnecessarily complicated and daunting to learn and manage relative to other open source components. As the ecosystem for database management evolves beyond the typical DBMS user base, opportunities are emerging for new programming models and new system components for data management and manipulation.

Architectural shifts in computing. While the variety of user scenarios is increasing, the computing substrates for data management are shifting dramatically as well. At the macro scale, the rise of cloud computing services suggests fundamental changes in software architecture. It democratizes access to parallel clusters of computers; every programmer has the opportunity and motivation to design systems and services that scale out incrementally to arbitrary degrees of parallelism. At a micro scale, computer architectures have shifted the focus of Moore's Law from increasing clock speed per chip to increasing the number of processor cores and threads per chip. In storage technologies, major changes are under way in the memory hierarchy due to the availability of more and larger on-chip caches, large inexpensive RAM, and flash memory. Power consumption has become an increasingly important aspect of the price/performance metric of large systems. These hardware trends alone motivate a wholesale reconsideration of data-management software architecture.

These factors together signal an urgent, widespread need for new data-management technologies. There is an opportunity for making a positive difference. Traditionally, the database community is known for the practical relevance of its research; relational databases are emblematic of technology transfer. But in recent years, the externally visible contribution of the database research community has not been as pronounced, and there is a mismatch between the notable expansion of the community's portfolio and its contribution to other fields of research and practice. In today's increasingly rich technical climate, the database community must recommit itself to impact and breadth. Impact is evaluated by external measures, so success involves helping new classes of users, powering new computing platforms, and making conceptual breakthroughs across computing. These should be the motivating goals for the next round of database research.

To achieve these goals, discussion at the 2008 Claremont Resort meeting revolved around two broad agendas we call reformation and synthesis. The reformation agenda involves deconstructing traditional data-centric ideas and systems and reforming them for new applications and architectural realities. One part of this entails focusing outside the traditional RDBMS stack and its existing interfaces, emphasizing new data-management systems for growth areas (such as e-science). Another part of the reformation agenda involves taking data-centric ideas like declarative programming and query optimization outside their original context in storage and retrieval to attack new areas of computing where a data-centric mindset promises to yield significant benefit. The synthesis agenda is intended to leverage research ideas in areas that have yet to develop identifiable, agreed-upon system architectures, including data integration, information extraction, and data privacy. Many of these subcommunities of database research seem ready to move out of the conceptual and algorithmic phase to work together on comprehensive artifacts (such as systems, languages, and services) that combine multiple techniques to solve complex user problems. Efforts toward synthesis can serve as rallying points for research, likely leading to new challenges and breakthroughs, and promise to increase the overall visibility of the work.

Research Opportunities

After two days of intense discussion at the 2008 Claremont meeting, it was surprisingly easy for the group to reach consensus on a set of research topics for investigation in coming years. Before exploring them, we stress a few points regarding what is not on the list. First, while we tried to focus on new opportunities, we do not propose they be pursued at the expense of existing good work. Several areas we deemed critical were left off because they are already focus topics in the database community. Many were mentioned in previous reports[1,3–7] and are the subject of significant efforts that require continued investigation and funding. Second, we kept the list short, favoring focus over coverage. Though most of us have other promising research topics we would have liked to discuss at greater length here, we focus on topics that

ILLUSTRATION BY GLUEKIT

attracted the broadest interest within the group.

In addition to the listed topics, the main issues raised during the meeting included management of uncertain information, data privacy and security, e-science and other scholarly applications, human-centric interaction with data, social networks and Web 2.0, personalization and contextualization of query- and search-related tasks, streaming and networked data, self-tuning and adaptive systems, and the challenges raised by new hardware technologies and energy constraints. Most are captured in the following discussion, with many cutting across multiple topics.

Revisiting database engines. System R and Ingres pioneered the architecture and algorithms of relational databases; current commercial databases are still based on their designs. But many of the changes in applications and technology demand a reformation of the entire system stack for data management. Current big-market relational database systems have well-known limitations. While they provide a range of features, they have only narrow regimes in which they provide peak performance; online transaction processing (OLTP) systems are tuned for lots of small, concurrent transactional debit/credit workloads, while decision-support systems are tuned for a few read-mostly, large-join-and-aggregation workloads. Meanwhile, for many popular data-intensive tasks developed over the past decade, relational databases provide poor price/performance and have been rejected; critical scenarios include text indexing, serving Web pages, and media delivery. New workloads are emerging in the sciences, Web 2.0-style applications, and other environments where database-engine technology could prove useful but is not bundled in current database systems.

Even within traditional application domains, the database marketplace today suggests there is room for significant innovation. For example, in the analytics markets for business and science, customers can buy petabytes of storage and thousands of processors, but the dominant commercial database systems typically cannot scale that far for many workloads. Even when they can, the cost of software and management relative to hardware is exorbitant. In the OLTP market, business imperatives like regulatory compliance and rapid response to changing business conditions raise the need to address data life-cycle issues (such as data provenance, schema evolution, and versioning).

The ubiquity of big data is expanding the base of users and developers of data-management technologies and will undoubtedly shake up the database research field.

Given these requirements, the commercial database market is wide open to new ideas and systems, as reflected in the recent funding climate for entrepreneurs. It is difficult to recall when there were so many start-up companies developing database engines, and the challenging economy has not trimmed the field much. The market will undoubtedly consolidate over time, but things are changing fast, and it remains a good time to try radical ideas.

Some research projects have begun taking revolutionary steps in database system architecture. There are two distinct directions: broadening the useful range of applicability for multipurpose database systems (for example, to incorporate streams, text search, XML, and information integration) and radically improving performance by designing special-purpose database systems for specific domains (for example, read-mostly analytics, streams, and XML). Both directions have merit, and the overlap in their stated targets suggests they may be more synergistic than not. Special-purpose techniques (such as new storage and compression formats) may be reusable in more general-purpose systems, and general-purpose architectural components (such as extensible query optimizer frameworks) may help speed prototyping of new special-purpose systems.

Important research topics in the core database engine area include:

- Designing systems for clusters of many-core processors that exhibit limited and nonuniform access to off-chip memory;
- Exploiting remote RAM and Flash as persistent media, rather than relying solely on magnetic disk;
- Treating query optimization and physical data layout as a unified, adaptive, self-tuning task to be carried out continuously;
- Compressing and encrypting data at the storage layer, integrated with data layout and query optimization;

- Designing systems that embrace nonrelational data models, rather than shoehorning them into tables;
- Trading off consistency and availability for better performance and thousands of machines; and
- Designing power-aware DBMSs that limit energy costs without sacrificing scalability.

This list is not exhaustive. One industrial participant at the Claremont meeting noted that this is a time of opportunity for academic researchers; the landscape has shifted enough that access to industrial legacy code provides little advantage, and large-scale clustered hardware is rentable in the cloud at low cost. Moreover, industrial players and investors are aggressively looking for bold new ideas. This opportunity for academics to lead in system design is a major change in the research environment.

Declarative programming for emerging platforms. Programmer productivity is a key long-acknowledged challenge in computing, with its most notable mention in the database context in Jim Gray's 1998 Turing lecture. Today, the urgency of the challenge is increasing exponentially as programmers target ever more complex environments, including many-core chips, distributed services, and cloud computing platforms.

Nonexpert programmers must be able to write robust code that scales out across processors in both loosely and tightly coupled architectures. Although developing new programming paradigms is not a database problem per se, ideas of data independence, declarative programming, and cost-based optimization provide a promising angle of attack. There is significant evidence that data-centric approaches will have significant influence on programming in the near term.

The recent popularity of the MapReduce programming framework for manipulating big data sets is an example of this potential. MapReduce is attractively simple, building on language and data-parallelism techniques that have been known for decades. For database researchers, the significance of MapReduce is in demonstrating the benefits of data-parallel programming to new classes of developers.

This is a unique opportunity for a fundamental "reformation" of the notion of data management, not as a single system but as a set of services that can be embedded, as needed, in many computing contexts.

This opens opportunities for the database community to extend its contribution to the broader community, developing more powerful and efficient languages and runtime mechanisms that help these developers address more complex problems.

As another example of declarative programming, in the past five years a variety of new declarative languages, often grounded in Datalog, have been developed for domain-specific systems in fields as diverse as networking and distributed systems, computer games, machine learning and robotics, compilers, security protocols, and information extraction. In many of these scenarios, the use of a declarative language has reduced code size by orders of magnitude while also enabling distributed or parallel execution. Surprisingly, the groups behind these efforts have coordinated very little with one another; the move to revive declarative languages in these new contexts has grown up organically.

A third example arises in enterprise-application programming. Recent language extensions (such as Ruby on Rails and LINQ) encourage query-like logic in programmer design patterns. But these packages have yet to address the challenge of enterprise-style programming across multiple machines; the closest effort here is DryadLINQ, focusing on parallel analytics rather than on distributed application development. For enterprise applications, a key distributed design decision is the partitioning of logic and data across multiple "tiers," including Web clients, Web servers, application servers, and a backend DBMS. Data independence is particularly valuable here, allowing programs to be specified without making a priori permanent decisions about physical deployment across tiers. Automatic optimization processes could make these decisions and move data and code as needed to achieve efficiency and correctness. XQuery has been proposed as an existing language that would facilitate this kind of declarative programming, in part because XML is often used in cross-tier protocols.

It is unusual to see this much energy surrounding new data-centric programming techniques, but the opportunity brings challenges as

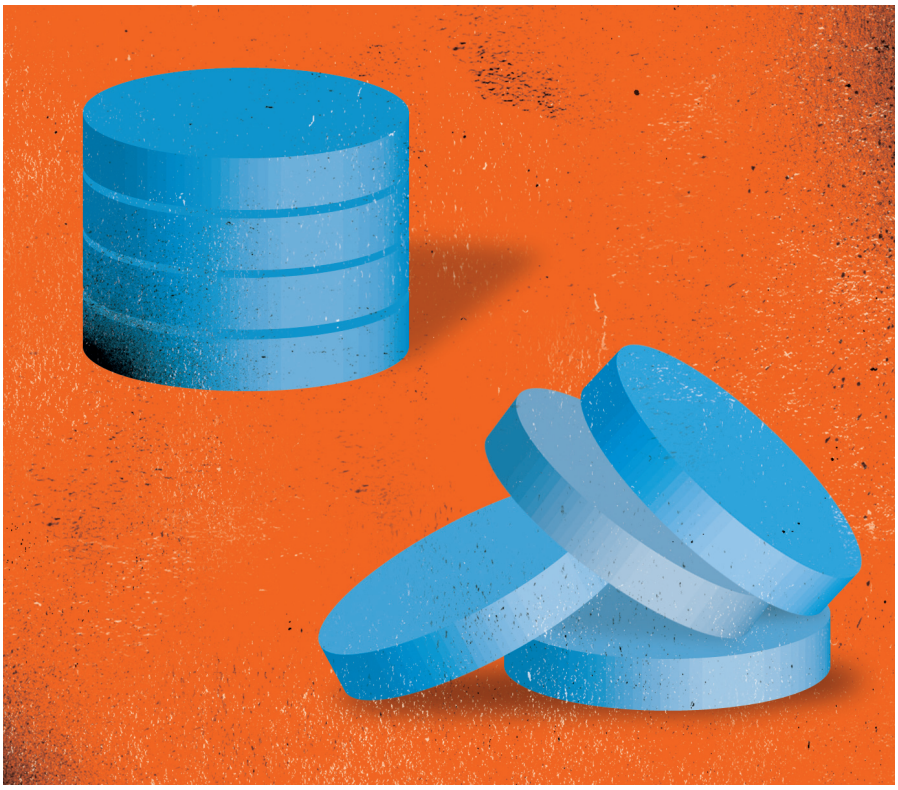
well. The research challenges include language design, efficient compilers and runtimes, and techniques to optimize code automatically across both the horizontal distribution of parallel processors and the vertical distribution of tiers. It seems natural that the techniques behind parallel and distributed databases—partitioned dataflow and cost-based query optimization—should extend to new environments. However, to succeed, these languages must be fairly expressive, going beyond simple MapReduce and select-project-join-aggregate dataflows. This agenda will require "synthesis" work to harvest useful techniques from the literature on database and logic programming languages and optimization, as well as to realize and extend them in new programming environments.

To genuinely improve programmer productivity, these new approaches also need to pay attention to the softer issues that capture the hearts and minds of programmers (such as attractive syntax, typing and modularity, development tools, and smooth interaction with the rest of the computing ecosystem, including networks, files, user interfaces, Web services, and other languages). This work also needs to consider the perspective of programmers who want to use their favorite programming languages and data services as primitives in those languages. Example code and practical tutorials are also critical.

To execute successfully, database research must look beyond its traditional boundaries and find allies throughout computing. This is a unique opportunity for a fundamental "reformation" of the notion of data management, not as a single system but as a set of services that can be embedded as needed in many computing contexts.

Interplay of structured and unstructured data. A growing number of data-management scenarios involve both structured and unstructured data. Within enterprises, we see large heterogeneous collections of structured data linked with unstructured data (such as document and email repositories). On the Web, we also see a growing amount of structured data primarily from three sources: millions of databases hidden behind forms (the deep Web); hundreds of millions of high-quality data items in HTML tables on Web pages and a growing number of mashups providing dynamic views on structured data; and data contributed by Web 2.0 services (such as photo and video sites, collaborative annotation services, and online structured-data repositories).

A significant long-term goal for the database community is to transition from managing traditional databases consisting of well-defined schemata for structured business data to the much more challenging task of managing a rich collection of structured, semi-structured, and unstructured data spread over many repositories in the enterprise and on the Web—sometimes referred to as the challenge of managing dataspaces.



In principle, this challenge is closely related to the general problem of data integration, a longstanding area for database research. The recent advances in this area and the new issues due to Web 2.0 resulted in significant discussion at the Claremont meeting. On the Web, the database community has contributed primarily in two ways: First, it developed technology that enables the generation of domain-specific ("vertical") search engines with relatively little effort; and second, it developed domain-independent technology for crawling through forms (that is, automatically submitting well-formed queries to forms) and surfacing the resulting HTML pages in a search-engine index. Within the enterprise, the database research community recently contributed to enterprise search and the discovery of relationships between structured and unstructured data.

The first challenge database researchers face is how to extract structure and meaning from unstructured and semistructured data. Information-extraction technology can now pull structured entities and relationships out of unstructured text, even in unsupervised Web-scale contexts. We expect in coming years that hundreds of extractors will be applied to a given data source. Hence developers and analysts need techniques for applying and managing predictions from large numbers of independently developed extractors. They also need algorithms that can introspect about the correctness of extractions and therefore combine multiple pieces of extraction evidence in a principled fashion. The database community is not alone in these efforts; to contribute in this area, database researchers should continue

ILLUSTRATION BY GLUEKIT

to strengthen ties with researchers in information retrieval and machine learning.

Context is a significant aspect of the semantics of the data, taking multiple forms (such as the text and hyperlinks that surround a table on a Web page, the name of the directory in which data is stored, accompanying annotations or discussions, and relationships to physically or temporally proximate data items). Context helps analysts interpret the meaning of data in such applications because the data is often less precise than in traditional database applications, as it is extracted from unstructured text, extremely heterogeneous, or sensitive to the conditions under which it was captured. Better database technology is needed to manage data in context. In particular, there is a need for techniques to discover data sources, enhance the data by discovering implicit relationships, determine the weight of an object's context when assigning it semantics, and maintain the provenance of data through these steps of storage and computation.

The second challenge is to develop methods for querying and deriving insight from the resulting sea of heterogeneous data. A specific problem is to develop methods to answer keyword queries over large collections of heterogeneous data sources. We must be able to break down the query to extract its intended semantics and route the query to the relevant sources(s) in the collection. Keyword queries are just one entry point into data exploration, and there is a need for techniques that lead users into the most appropriate querying mechanism. Unlike previous work on information integration, the challenges here are that we cannot assume we have semantic mappings for the data sources and we cannot assume that the domain of the query or the data sources is known. We need to develop algorithms for providing best-effort services on loosely integrated data. The system should provide meaningful answers to queries with no need for manual integration and improve over time in a pay-as-you-go fashion as semantic relationships are discovered and refined. Developing index structures to support querying hybrid data is also a significant challenge. More generally, we need to develop new notions of correctness and consistency in order to provide metrics and enable users or system designers to make cost/quality trade-offs. We also need to develop the appropriate systems concepts around which these functionalities are tied.

In addition to managing existing data collections, there is an opportunity to innovate in the creation of data collections. The emergence of Web 2.0 creates the potential for new kinds of data-management scenarios in which users join ad hoc communities to create, collaborate, curate, and discuss data online. As an example, consider creating a database of access to clean water in different places around the world. Since such communities rarely agree on schemata ahead of time, the schemata must be inferred from the data; however, the resulting schemata are still used to guide users to consensus. Systems in this context must incorporate visualizations that drive exploration and analysis. Most important, these systems must be extremely easy to use and so will probably require compromising on some typical database functionality and providing more semiautomatic "hints" mined from the data. There is an important opportunity for a feedback loop here; as more data is created with such tools, information extraction and querying could become easier. Commercial and academic prototypes are beginning to appear, but there is plenty of room for additional innovation and contributions.

Cloud data services. Economic and technological factors have motivated a resurgence of shared computing infrastructure, providing software and computing facilities as a service, an approach known as cloud services or cloud computing. Cloud services provide efficiencies for application providers by limiting up-front capital expenses and by reducing the cost of ownership over time. Such services are typically hosted in a data center using shared commodity hardware for computation and storage. A varied set of cloud services is available today, including application services (salesforce.com), storage services (Amazon S3), compute services (Amazon EC2, Google App Engine, and Microsoft Azure), and data services (Amazon SimpleDB, Microsoft SQL Data Services, and Google's Datastore). They represent a major reformation of data-management architectures, with more on the horizon. We anticipate many future data-centric applications lever-

ILLUSTRATION BY GLUEKIT

aging data services in the cloud.

A cross-cutting theme in cloud services is the trade-off providers face between functionality and operational costs. Today's early cloud data services offer an API that is much more restricted than that of traditional database systems, with a minimalist query language, limited consistency guarantees, and in some cases explicit constraints on resource utilization. This limited functionality pushes more programming burden on developers but allows cloud providers to build more predictable services and offer service-level agreements that would be difficult to provide for a full-function SQL data service. More work and experience are needed on several fronts to fully understand the continuum between today's early cloud data services and more full-function but possibly less-predictable alternatives.

Manageability is particularly important in cloud environments. Relative to traditional systems, it is complicated by three factors: limited human intervention, high-variance workloads, and a variety of shared infrastructures. In the majority of cloud-computing settings, there will be no database administrators or system administrators to assist developers with their cloud-based applications; the platform must do much of that work automatically. Mixed workloads have always been difficult to tune but may be unavoidable in this context.

Even a single customer's workload can vary widely over time; the elastic provisioning of cloud services makes it economical for a user to occasionally harness orders-of-magnitude more resources than usual for short bursts of work. Meanwhile, service tuning depends heavily on the way the shared infrastructure is "virtualized." For example, Amazon EC2 uses hardware-level virtual machines as its programming interface. On the opposite end of the spectrum, salesforce.com implements "multi-tenant" hosting of many independent schemas in a single managed DBMS. Many other virtualization solutions are possible, each with different views into the workloads above and platforms below and different abilities to control each. These variations require revisiting traditional roles and responsibilities for resource management across layers.

Limited functionality pushes more programming burden on developers but allows cloud providers to build more predictable services and offer service-level agreements that would be difficult to provide for a full-function SQL data service.

The need for manageability adds urgency to the development of self-managing database technologies that have been explored over the past decade. Adaptive, online techniques will be required to make these systems viable, while new architectures and APIs, including the flexibility to depart from traditional SQL and transactional semantics when prudent, reduce requirements for backward compatibility and increase the motivation for aggressive redesign.

The sheer scale of cloud computing involves its own challenges. Today's SQL databases were designed in an era of relatively reliable hardware and intensive human administration; as a result, they do not scale effectively to thousands of nodes being deployed in a massively shared infrastructure. On the storage front, it is unclear whether these limitations should be addressed with different transactional implementation techniques, different storage semantics, or both simultaneously. The database literature is rich in proposals on these issues. Cloud services have begun to explore simple pragmatic approaches, but more work is needed to synthesize ideas from the literature in modern cloud computing regimes. In terms of query processing and optimization, it will not be feasible to exhaustively search a domain that considers thousands of processing sites, so some limitations on either the domain or the search will be required. Finally, it is unclear how programmers will express their programs in the cloud, as discussed earlier.

The sharing of physical resources in a cloud infrastructure puts a premium on data security and privacy that cannot be guaranteed by physical boundaries of machines or networks. Hence cloud services are fertile ground for efforts to synthesize and accelerate the work the database community has done in these areas. The key to success is to specifically target usage scenarios in the cloud, seated in practical economic incentives for service providers and customers.

As cloud data services become popular, new scenarios will emerge with their own challenges. For example, we anticipate specialized services that are pre-loaded with large data sets (such as

stock prices, weather history, and Web crawls). The ability to "mash up" interesting data from private and public domains will be increasingly attractive and provide further motivation for the challenges discussed earlier concerning the interplay of structured and unstructured data. The desire to mash up data also points to the inevitability of services reaching out across clouds, an issue already prevalent in scientific data "grids" that typically have large shared data servers at multiple sites, even within a single discipline. It also echoes, in the large, the standard proliferation of data sources in most enterprises. Federated cloud architectures will only add to these challenges.

Mobile applications and virtual worlds. This new class of applications, exemplified by mobile services and virtual worlds, is characterized by the need to manage massive amounts of diverse user-created data, synthesize it intelligently, and provide real-time services. The database community is beginning to understand the challenges faced by these applications, but much more work is needed. Accordingly, the discussion about these topics at the meeting was more speculative than about those of the earlier topics but still deserve attention.

Two important trends are changing the nature of the field. First, the platforms on which mobile applications are built—hardware, software, and network—have attracted large user bases and ubiquitously support powerful interactions "on the go." Second, mobile search and social networks suggest an exciting new set of mobile applications that can deliver timely information (and advertisements) to mobile users depending on location, personal preferences, social circles, and extraneous factors (such as weather), as well as the context in which they operate. Providing these services requires synthesizing user input and behavior from multiple sources to determine user location and intent.

The popularity of virtual worlds like Second Life has grown quickly and in many ways mirrors the themes of mobile applications. While they began as interactive simulations for multiple users, they increasingly blur the distinctions with the real world and suggest the potential for a more data-rich mix. The term "co-space" is sometimes used to refer to a coexisting space for both virtual and physical worlds. In it, locations and events in the physical world are captured by a large number of sensors and mobile devices and materialized within a virtual world. Correspondingly, certain actions or events within the virtual world affect the physical world (such as shopping, product promotion, and experiential computer gaming). Applications of co-space include rich social networking, massive multi-player games, military training, edutainment, and knowledge sharing.

Electronic media underscore the modern reality that it is easy to be widely published but much more difficult to be widely read.

In both areas, large amounts of data flow from users and get synthesized and used to affect the virtual and/or real world. These applications raise new challenges, including how to process heterogeneous data streams in order to materialize real-world events, how to balance privacy against the collective benefit of sharing personal real-time information, and how to apply more intelligent processing to send interesting events in the co-space to someone in the physical world.

The programming of virtual actors in games and virtual worlds requires large-scale parallel programming; declarative methods have been proposed as a solution in this environment, as discussed earlier. These applications also require development of efficient systems, as suggested earlier in the context of database engines, including appropriate storage and retrieval methods, data-processing engines, parallel and distributed architectures, and power-sensitive software techniques for managing the events and communications across large number of concurrent users.

Moving Forward

The 2008 Claremont meeting also involved discussions on the database research community's processes, including organization of publication procedures, research agendas, attraction and mentorship of new talent, and efforts to ensure a benefit from the research on practice and toward furthering our understanding of the field. Some of the trends seen in database research are echoed in other areas of computer science. Whether or not they are, the discussion may be of broader interest in the field.

Prior to the meeting, a team led by one of the participants performed a bit of ad hoc data analysis over database conference bibliographies from the DBLP repository (dblp.uni-trier.de). While the effort was not scientific, the results indicated that the database research community has doubled in size over the past decade, as suggested by several metrics: number of published papers, number of distinct authors, number of distinct institutions to which these authors belong, and number of session topics at conferences, loosely defined. This served as a backdrop to the discussion that followed. An open question is whether this phenomenon is emerging at larger scales—in computer science and in science in general. If so, it may be useful to discuss the management of growth at those larger scales.

The growth of the database community puts pressure on the content and processes of database research publications. In terms of content, the increasingly technical scope of the community makes it difficult for individual researchers to keep track of the field. As a result, survey articles and tutorials are increasingly important to the community. These efforts should be encouraged informally within the community, as well as via professional incentive structures (such as academic tenure and promotion in industrial labs). In terms of processes, the reviewing load for papers is increasingly burdensome, and there was a perception at the Claremont meeting that the quality of reviews had been decreasing. It was suggested at the meeting that the lack of face-to-face program-committee meetings in recent years has exacerbated the problem of poor reviews and removed opportunities for risky or speculative papers to be championed effectively over well-executed but more pedestrian work.

There was some discussion at the meeting about recent efforts—notably by ACM-SIGMOD and VLDB—to enhance the professionalism of papers and the reviewing process via such mechanisms as double-blind reviewing and techniques to encourage experimental repeatability. Many participants were skeptical that the efforts to date have contributed to long-term research quality, as measured in intellectual and practical relevance. At the same time, it was acknowledged that the database community's growth increases the need for clear and clearly enforced processes for scientific publication. The challenge going forward is to find policies that simultaneously reward big ideas and risk-taking while providing clear and fair rules for achieving these rewards. The publication venues would do well to focus as much energy on processes to encourage relevance and innovation as they do on processes to encourage rigor and discipline.

In addition to tuning the mainstream publication venues, there is an opportunity to take advantage of other channels of communication. For example, the database research community has had little presence in the relatively active market for technical books. Given the growing population of developers working with big data sets, there is a need for accessible books on scalable data-management algorithms and techniques that programmers can use to build software. The current crop of college textbooks is not targeted at this market. There is also an opportunity to present database research contributions as big ideas in their own right, targeted at intellectually curious readers outside the specialty. In addition to books, electronic media (such as blogs and wikis) can complement technical papers by opening up different stages of the research life cycle to discussion, including status reports on ongoing projects, concise presentation of big ideas, vision statements, and speculation. Online fora can also spur debate and discussion if appropriately provocative. Electronic media underscore the modern reality that it is easy to be widely published but much more difficult to be widely read. This point should be reflected in the mainstream publication context, as well as by authors and reviewers. In the end, the consumers of an idea define its value.

Given the growth in the database research community, the time is ripe for ambitious projects to stimulate collaboration and cross-fertilization of ideas. One proposal is to foster more data-driven research by building a globally shared collection of structured data, accepting contributions from all parties. Unlike previous efforts in this vein, the collection should not be designed for any particular benchmark; in fact, it is likely that most of the interesting problems suggested by this data are as yet unidentified.

There was also discussion at the meeting of the role of open source software development in the database community. Despite a tradition of open source software, academic database researchers have only rarely reused or shared software. Given the current climate, it might be useful to move more aggressively toward sharing software and collaborating on software projects across institutions. Information integration was mentioned as an area in which such an effort is emerging.

Finally, interest was expressed in technical competitions akin to the Netflix Prize (www.netflixprize.com) and KDD Cup (www.sigkdd.org/kddcup/index.php) competitions. To kick off this effort in the database domain, meeting participants identified two promising areas for competitions: system components for cloud computing (likely measured in terms of efficiency) and large-scale information extraction (likely measured in terms of accuracy and efficiency). While it was noted that each of these proposals requires a great deal of time and care to realize, several participants volunteered to initiate efforts. That work has begun with the 2009 SIGMOD Programming Contest (db.csail.mit.edu/sigmod09contest). C

References

1. Abiteboul, S. et al. The Lowell database research self assessment. *Commun. ACM 48*, 5 (May 2005), 111–118.
2. Austin, I. I.B.M. acquires Cognos, maker of business software, for $4.9 billion. *New York Times* (Nov. 11, 2007).
3. Bernstein, P.A. et al. The Asilomar report on database research. *SIGMOD Record 27*, 4 (Dec. 1998), 74–80.
4. Bernstein, P.A. et al. Future directions in DBMS research: The Laguna Beach participants. *SIGMOD Record 18*, 1 (Mar. 1989), 17–26.
5. Silberschatz, A. and Zdonik, S. Strategic directions in database systems: Breaking out of the box. *ACM Computing Surveys 28*, 4 (Dec. 1996), 764–778.
6. Silberschatz, A., Stonebraker, M., and Ullman, J.D. Database research: Achievements and opportunities into the 21st century. *SIGMOD Record 25*, 1 (Mar. 1996), 52-63.
7. Silberschatz, A., Stonebraker, M., and Ullman, J.D. Database systems: Achievements and opportunities. *Commun. ACM 34*, 10 (Oct. 1991), 110–120.

Correspondence regarding this article should be addressed to **Joseph M. Hellerstein** (hellerstein@cs.berkeley.edu).