DOI:10.1145/2845915

DANIEL ABADI
RAKESH AGRAWAL
ANASTASIA AILAMAKI
MAGDALENA BALAZINSKA
PHILIP A. BERNSTEIN
MICHAEL J. CAREY
SURAJIT CHAUDHURI
JEFFREY DEAN
ANHAI DOAN
MICHAEL J. FRANKLIN
JOHANNES GEHRKE
LAURA M. HAAS
ALON Y. HALEVY
JOSEPH M. HELLERSTEIN
YANNIS E. IOANNIDIS
H.V. JAGADISH
DONALD KOSSMANN
SAMUEL MADDEN
SHARAD MEHROTRA
TOVA MILO
JEFFREY F. NAUGHTON
RAGHU RAMAKRISHNAN
VOLKER MARKL
CHRISTOPHER OLSTON
BENG CHIN OOI
CHRISTOPHER RÉ
DAN SUCIU
MICHAEL STONEBRAKER
TODD WALTER
JENNIFER WIDOM

Database researchers paint big data as a defining challenge. To make the most of the enormous opportunities at hand will require focusing on five research areas.

The Beckman Report on Database Research

A GROUP OF database researchers meets periodically to discuss the state of the field and its key directions going forward. Past meetings were held in 1989,[6] 1990,[11] 1995,[12] 1996,[10] 1998,[7] 2003,[1] and 2008.[2] Continuing this tradition, 28 database researchers and two invited speakers met in October 2013 at the Beckman Center on the University of California-Irvine campus for two days of discussions. The meeting attendees represented a broad cross-section of interests, affiliations, seniority, and geography. Attendance was capped at 30 so the meeting would be as interactive as possible. This article summarizes the conclusions from that meeting; an extended report and participant presentations are available at http://beckman.cs.wisc.edu.

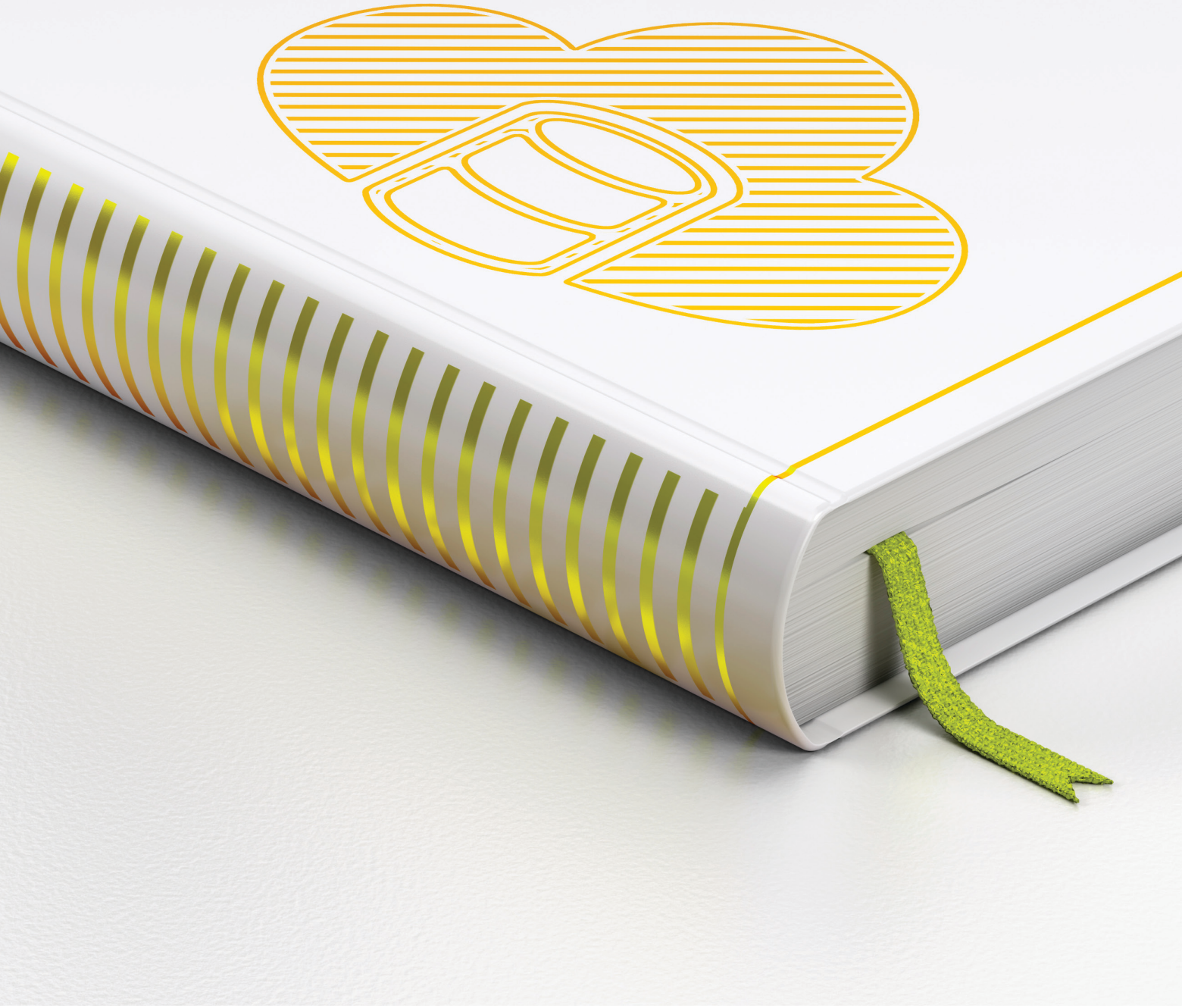

IMAGE BY MAKSIM KABAKOU

The meeting participants quickly converged on big data as a defining challenge of our time. Big data arose due to the confluence of three major trends. First, it has become much cheaper to generate a wide variety of data, due to inexpensive storage, sensors, smart devices, social software, multiplayer games, and the Internet of Things, which connects homes, cars, appliances, and other devices. Second, it has become much cheaper to process large amounts of data, due to advances in multicore CPUs, solid state storage, inexpensive cloud computing, and open source software. Finally, data management has become democratized. The process of generating, processing, and consuming data is no longer just for database professionals. Decision makers, domain scientists, application users, journalists, crowd workers, and everyday consumers now routinely do it.

Due to these trends, an unprecedented volume of data needs to be captured, stored, queried, processed, and turned into knowledge. These goals are remarkably well aligned with those that have driven the database research community for decades. Many early systems for big data abandoned database management system (DBMS) principles, such as declarative programming and transactional data consistency, in favor of scalability and

» key insights

- **Thirty leaders from the database research community met in October 2013 to discuss the state of the field and important future research directions.**
- **Big data was identified as a defining challenge for the field. Five related challenges were called out: developing scalable data infrastructures, coping with increased diversity in both data and data management, addressing the end-to-end data-to-knowledge pipeline, responding to the adoption of cloud-based computing, and accomodating the many and changing roles of individuals in the data life cycle.**
- **College-level database education needs modernization to catch up with the many changes in database technology of the past decade and to meet the demands of the emerging disciplines of data science.**

fault tolerance on commodity hardware. However, the latest generation of big data systems is rediscovering the value of these principles and is adopting concepts and methods that have been long-standing assets of the database community. Building on these principles and assets, the database community is well positioned to drive transformative improvements to big data technology.

But big data also brings enormous challenges, whose solutions will require massive disruptions to the design, implementation, and deployment of data management solutions. The main characteristics of big data are volume, velocity, and variety. The database community has worked on volume and velocity for decades, and has developed solutions that are mission critical to virtually every commercial enterprise on the planet. The unprecedented scale of big data, however, will require a radical rethinking of existing solutions.

Variety arises from several sources. First, there is the problem of integrating and analyzing data that comes from diverse sources, with varying formats and quality. This is another long-standing topic of database work, yet it is still an extremely labor-intensive journey from raw data to actionable knowledge. This problem is exacerbated by big data, causing a major bottleneck in the data processing pipeline. Second, there is the variety of computing platforms needed to process big data: hardware infrastructures; processing frameworks, languages, and systems; and programming abstractions. Finally, there is a range of user sophistication and preferences. Designing data management solutions that can cope with such extreme variety is a difficult challenge.

Moving beyond the three Vs, many big data applications will be deployed in the cloud, both public and private, on a massive scale. This requires new techniques to offer predictable performance and flexible interoperation. Many applications will also require people to solve semantic problems that still bedevil current automatic solutions. This can range from a single domain expert to a crowd of workers, a user community, or the entire connected world (for example, Wikipedia). This will require new techniques to help people be more productive and to reduce the skill level needed to solve these problems.

Many big data applications will be deployed in the cloud, both public and private, on a massive scale. This requires new techniques to offer predictable performance and flexible interoperation.

Finally, big data brings important community challenges. We must rethink the approach to teaching data management, reexamine our research culture, and adapt to the emergence of data science as a discipline.

Research Challenges

The meeting identified five big data challenges: scalable big/fast data infrastructures; coping with diversity in data management; end-to-end processing of data; cloud services; and the roles of people in the data life cycle. The first three challenges deal with the volume, velocity, and variety aspects of big data. The last two deal with deploying big data applications in the cloud and managing the involvement of people in these applications.

These big data challenges are not an exclusive agenda to be pursued at the expense of existing work. In recent years the database community has strengthened core competencies in relational DBMSs and branched out into many new directions. Some important issues raised repeatedly during the meeting are security, privacy, data pricing, data attribution, social and mobile data, spatiotemporal data, personalization and contextualization, energy-constrained processing, and scientific data management. Many of these issues cut across the identified big data challenges and are captured in the discussion here.

It is important to note that some of this work is being done in collaboration with other computer science fields, including distributed systems, artificial intelligence, knowledge discovery and data mining, human-computer interaction, and e-science. In many cases, these fields provided the inspiration for the topic and the data management community has joined in, applying its expertise to produce robust solutions. These collaborations have been very productive and should continue to grow.

Scalable big/fast data infrastructures. *Parallel and distributed processing.* In the database world, parallel processing of large structured datasets has been a major success, leading to several generations of SQL-based

products that are widely used by enterprises. Another success is data warehousing, where database researchers defined the key abstraction of data cube (for online analytic processing, or OLAP) and strategies for querying it in parallel, along with support for materialized views and replication. The distributed computing field has achieved success in scaling up data processing for less structured data on large numbers of unreliable, commodity machines using constrained programming models such as MapReduce. Higher-level languages have been layered on top, to enable a broader audience of developers to use scalable big data platforms. Today, open source platforms such as Hadoop[3]—with its MapReduce programming model, large-scale distributed file system, and higher-level languages, such as Pig[5] and Hive[4]—are seeing rapid adoption for processing less structured data, even in traditional enterprises.

Query processing and optimization. Given the enthusiastic adoption of declarative languages for processing big data, there is a growing recognition that more powerful cost-aware query optimizers and set-oriented query execution engines are needed, to fully exploit large clusters of many-core processors, scaling both "up" and "out." This will create challenges for progress monitoring, so a user can diagnose and manage queries that are running too slowly or consuming excessive resources. To adapt to the characteristics of previously unseen data and reduce the cost of data movement between stages of data analysis, query processors will need to integrate data sampling, data mining, and machine learning into their flows.

New hardware. At datacenter scale, the ratio between the speed of sequential processing and interconnects is changing with the advent of faster networks, full bisection bandwidth networks between servers, and remote direct memory access. In addition to clusters of general-purpose multicore processors, more specialized processors should be considered. Commercially successful database machines have shown the potential of hardware-software co-design for data management. Researchers should continue to explore ways of leveraging specialized processors, for example, graphics processing units, field-programmable gate arrays, and application-specific integrated circuits, for processing very large datasets. These changes in communications and processing technologies will require a reconsideration of parallel and distributed query-processing algorithms, which have traditionally focused on more homogeneous hardware environments.

Cost-efficient storage. The database research community must learn how best to leverage emerging memory and storage technologies. Relative to commodity magnetic disks, solid-state disks are expensive per gigabyte but cheap per I/O operation. Various non-volatile random-access memory technologies are under development, all with different speed, power, and durability characteristics.

Both server-attached and network-attached storage architectures need to be considered. Distributed file systems like HDFS, which are server-attached yet shared across the network, are a hybrid of both approaches. How best to use this range of storage configurations reopens many questions reminiscent of past debates of shared memory vs. shared disk vs. shared nothing, questions many have considered to be "closed" for parallel relational systems.

High-speed data streams. For data that arrives at ever-higher speeds, new scalable techniques for ingesting and processing streams of data will be needed. Algorithms will need to be tuned carefully to the behavior of hardware, for example, to cope with non-uniform memory access and limited transfer rates across layers of the memory hierarchy. Some very high-speed data sources, often with lower information density, will need to be processed online and then discarded without being persisted in its entirety. Rather, samples and aggregations of such data will need to be selected and stored persistently to answer queries that arrive after the raw data is no longer available. For such data, progressive query processing will be important to provide incremental and partial results with increasing accuracy as data flows through the processing pipeline.

Late-bound schemas. For data that is persisted but processed just once (if ever), it makes little sense to pay the substantial price of storing and indexing it first in a database system. Instead, it should be stored as a binary file and interpreted as a structured record only if and when it is read later. Record structure may be self-describing via attribute-value pairs, such as JavaScript Object Notation (JSON), interpreted via predefined schemas, or deduced using data mining. To offer the benefits of database queries in such scenarios, we need query engines that can run efficiently over raw files with late-bound schemas.

Consistency. Today's world brings new requirements for data capture, updates, and simple and fast data access. Handling high rates of data capture and updates for schema-less data has led to the development of NoSQL systems. There are many such systems, with a range of transaction models. Most provide only basic data access and weak atomicity and isolation guarantees, making it difficult to build and reason about reliable applications. As a result, a new class of big data system has emerged that provides full-fledged database-like features over key-value stores or similar substrates. For some applications, the stored data is still managed and updated as "the source of truth" for an enterprise. For others, such as the Internet of Things, the stored data reflects ongoing events in the outside world that applications can use to recognize and respond to situations of interest. This creates an opportunity to revisit programming models and mechanisms for data currency and consistency and to design new models and techniques for developing robust applications.

Metrics and benchmarks. Finally, scalability should be measured not only in petabytes of data and queries per second, but also total cost of ownership (including management and energy use), end-to-end processing speed (that is, time from raw data arrival to eventual insights), brittleness (for example, the ability to continue despite failures such as partial data parse errors), and usability (especially for entry-level users). To measure progress against such broader metrics, new

types of benchmarks will be required.

Diversity in data management. *No one-size-fits-all.* Today's data-driven world involves a richer variety of data types, shapes, and sizes than traditional enterprise data, which is stored in a data warehouse optimized for analysis tasks. Today, data is often stored in different representations managed by different software systems with different application programming interfaces, query processors, and analysis tools. It seems unlikely a single, one-size-fits-all, big data system will suffice for this degree of diversity. Instead, we expect multiple classes of systems to emerge, each addressing a particular need (for example, data deduplication, analysis of large graphs, diverse scientific experiments, and real-time stream processing) or exploiting a particular type of hardware platform (for example, clusters of inexpensive machines or large multicore servers). Addressing these scenarios will require applying expertise in set-oriented parallel processing and in efficiently handling datasets that do not fit in main memory.

Cross-platform integration. Given this diversity of systems, platforms will need to be integrated or federated to enable data analysts to combine and analyze data across systems. This will involve not only hiding the heterogeneity of data formats and access languages, but also optimizing the performance of accesses that span diverse big data systems and of flows that move data between them. It will also require managing systems that run on diverse devices and span large datacenters. Disconnected devices will become increasingly common, raising challenges in reliable data ingestion, query processing, and data inconsistency in such sometimes-connected, wide-area environments.

Programming models. A diverse and data-driven world requires diverse programming abstractions to operate on very large datasets. A single data analysis language for big data, such as an extension of SQL, will not meet everyone's needs. Rather, users must be able to analyze their data in the idiom they find most natural: SQL, Pig, R, Python, a domain-specific language, or a lower-level constrained programming model such as MapReduce or Valiant's bulk synchronous processing model. This also suggests the development of reusable middle-layer components that can support multiple language-specific bindings, such as scalable support for matrix multiplication, list comprehension, and stylized iterative execution models. Another potentially fruitful focus is tools for the rapid development of new domain-specific data analysis languages—tools that simplify the implementation of new scalable, data-parallel languages.

Data processing workflows. To handle data diversity, we need platforms that can span both "raw" and "cooked" data. The cooked data can take many forms, for example, tables, matrices, or graphs. Systems will run end-to-end workflows that mix multiple types of data processing, for example, querying data with SQL and then analyzing it with R. To unify diverse systems, lazy computation is sometimes beneficial—lazy data parsing, lazy conversion and loading, lazy indexing and view construction, and just-in-time query planning. Big data systems should become more interoperable like "Lego bricks." Cluster resource managers, such as Hadoop 2.0's YARN, provide some inspiration at the systems level, as do workflow systems for the Hadoop ecosystem and tools for managing scientific workflows.

End-to-end processing of data. The database research community should pay more attention to end-to-end processing of data. Despite years of R&D, surprisingly few tools can go from raw data all the way to extracted knowledge without significant human intervention at each step. For most steps, the intervening people need to be highly computer savvy.

Data-to-knowledge pipeline. The steps of the raw-data-to-knowledge pipeline will be largely unchanged: data acquisition; selection, assessment, cleaning, and transformation (also called "data wrangling"); extraction and integration; mining, OLAP, and analytics; and result summarization, provenance, and explanation. In addition to greater scale, what has significantly changed is the greater diversity of data and users. Data today comes in a wide variety of formats. Often, structured and unstructured data must be used together in a structured fashion. Data tools must exploit human feedback in every step of the analytical pipeline, and must be usable by subject-matter experts, not just by IT professionals. For example, a journalist may want to clean, map, and publish data from a spreadsheet file of crime statistics. Tools must also be tailored to data scientists, the new class of data analysis professionals that has emerged.

Tool diversity. Since no one-size-fits-all tool will cover the wide variety of data analysis scenarios ahead, we need multiple tools, each solving a step of the raw-data-to-knowledge pipeline. They must be seamlessly integrated and easy to use for both lay and expert users, with best-practice guidance on when to use each tool.

Tool customizability. Tools should be able to exploit domain knowledge, such as dictionaries, knowledge bases, and rules. They should be easy to customize to a new domain, possibly using machine learning to automate the customization process. Handcrafted rules will remain important, though, as many analysis applications require very high precision, such as e-commerce. For such applications, analysts often write many rules to cover "corner cases" that are not amenable to learning and generalization. Thus, tools should provide support for writing, evaluating, applying, and managing handcrafted rules.

Open source. Few tools in this area are open source. Most are expensive proprietary products that address certain processing steps. As a result, existing tools cannot easily benefit from ongoing contributions by the data integration research community.

Understanding data. Explanation, provenance, filtering, summarization, and visualization requirements will be critical to making analytic tools easy to use. Capturing and managing appropriate meta-information is key to enable explanation, provenance, reuse, and visualization. Visual analytics is receiving growing attention in the database, visualization, and HCI communities. Continued progress in this area is essential to help users cope with big data volumes.

Knowledge bases. The more knowledge we have about a target domain,

the better that tools can analyze the domain. As a result, there has been a growing trend to create, share, and use domain knowledge to better understand data. Such knowledge is often captured in knowledge bases (KBs) that describe the most important entities and relationships in a domain, such as a KB containing profiles of tens of thousands of biomedical researchers along with their publications, affiliations, and patents. Such KBs are used for improving the accuracy of the raw-data-to-knowledge pipeline, answering queries about the domain, and finding domain experts. Many companies have also built KBs for answering user queries, annotating text, supporting e-commerce, and analyzing social media. The KB trend will likely accelerate, leading to a proliferation of community-maintained "knowledge centers" that offer tools to query, share, and use KBs for data analysis.

While some progress has been made on this topic, more work is needed on tools to help groups of users with different skill levels collaboratively build, maintain, query, and share domain-specific KBs.

Cloud services. Cloud computing comes in three main forms: Infrastructure as a Service (IaaS), where the service is virtualized hardware; Platform as a Service (PaaS), where the service is virtualized infrastructure software such as a DBMS; and Software as a Service (SaaS), where the service is a virtualized application such as a customer relationship management solution. From a data platform perspective, the ideal goal is a PaaS for data, where users can upload data to the cloud, query it as they do today over their on-premise SQL databases, and selectively share the data and results easily, all without worrying about how many instances to rent, what operating system to run on, how to partition databases across servers, or how to tune them. Despite the emergence of services such as Database.com from Salesforce.com, Big Query from Google, Redshift from Amazon, and Azure SQL Database from Microsoft, we have yet to achieve the full ideal. Here, we outline some of the critical challenges to realize the complete vision of a Data PaaS in the cloud.

Elasticity. Data can be prohibitively expensive to move. Network-attached storage makes it easier to scale out a database engine. However, network latency and bandwidth limit database performance. Server-attached storage reduces these limitations, but then server failures can degrade availability and failover can interfere with load balancing and hence violate service-level agreements (SLAs).

A diverse and data-driven world requires diverse programming abstractions to operate on very large datasets.

An open question is whether the same cloud storage service can support both transactions and analytics; how caching best fits into the overall picture is also unclear. To provide elasticity, database engines and analysis platforms in a Data PaaS will need to operate well on top of resources that can be allocated quickly during workload peaks but possibly preempted for users paying for premium service.

Data replication. Latency across geographically distributed datacenters makes it difficult to keep replicas consistent yet offer good throughput and response time to updates. Multi-master replication is a good alternative, when conflicting updates on different replicas can be automatically synchronized. But the resulting programming model is not intuitive to mainstream programmers. Thus, the challenge is how best to trade-off availability, consistency performance, programmability, and cost.

System administration and tuning. In the world of Data PaaS, database and system administrators simply do not exist. Therefore, all administrative tasks must be automated, such as capacity planning, resource provisioning, and physical data management. Resource control parameters must also be set automatically and be highly responsive to changes in load, such as buffer pool size and admission control limits.

Multitenancy. To be competitive, a Data PaaS should be cheaper than an on-premises solution. This requires providers to pack multiple tenants together to share physical resources to smooth demand and reduce cost. This introduces several problems. First, the service must give security guarantees against information leakage across tenants. This can be done by isolating user databases in separate files and running the database engine in separate virtual machines (VMs). However,

this is inefficient for small databases, and makes it difficult to balance resources between VMs running on the same server. An alternative is to have users share a single database and database engine instance. But then special care is needed to prevent cross-tenant accesses. Second, users want an SLA that defines the level of performance and availability they need. Data PaaS providers want to offer SLAs too, to enable tiered pricing. However, it is challenging to define SLAs that are understandable to users and implementable by PaaS providers. The implementation challenge is to ensure performance isolation between tenants, to ensure a burst of demand from one tenant does not cause a violation of other tenants' SLAs.

Data sharing. The cloud enables sharing at an unprecedented scale. One problem is how to support essential services such as data curation and provenance collaboratively in the cloud. Other problems include: how to find useful public data, how to relate self-managed private data with public data to add context, how to find high-quality data in the cloud, how to share data at fine-grained levels, how to distribute costs when sharing computing and data, and how to price data. The cloud also creates new life-cycle challenges, such as how to protect data if the current cloud provider fails and to preserve data for the long term when users who need it have no personal or financial connection to those who provide it. The cloud will also drive innovation in tools for data governance, such as auditing, enforcement of legal terms and conditions, and explanation of user policies.

Hybrid clouds. There is a need for interoperation of database services among the cloud, on-premise servers, and mobile devices. One scenario is off-loading. For example, users may run applications in their private cloud during normal operation, but tap into a public cloud at peak times or in response to unanticipated workload surges. Another is cyber-physical systems, such as the Internet of Things. For example, cars will gather local sensor data, upload some of it into the cloud, and obtain control information in return based on data aggregation from many sources. Cyber-physical systems involve data streaming from multiple sensors and mobile devices, and must cope with intermittent connectivity and limited battery life, which pose difficult challenges for real-time and perhaps mission-critical data management in the cloud.

We need to build platforms that allow people to curate data easily and extend relevant applications to incorporate such curation.

Roles of humans in the data life cycle. Back when data management was an enterprise-driven activity, roles were clear: developers built databases and database-centric applications, business analysts queried databases using (SQL-based) reporting tools, end users generated data and queried and updated databases, and database administrators tuned and monitored databases and their workloads. Today, a single individual can play multiple roles in the data life cycle, and some roles may be served by crowdsourcing. Thus, human factors need to be considered for query understanding and refinement, identifying relevant and trustworthy information sources, defining and incrementally refining the data processing pipeline, visualizing relevant patterns, obtaining query answers, and making the various microtasks doable by domain experts and end users. We can classify people's roles into four general categories: producers, curators, consumers, and community members.

Data producers. Today, virtually anyone can generate a torrent of data from mobile phones, social platforms and applications, and wearable devices. One key challenge for the database community is to develop algorithms and incentives that guide people to produce and share the most useful data, while maintaining the desired level of data privacy. When people produce data, how can we help them add metadata quickly and accurately? For example, when a user uploads an image, Facebook automatically identifies faces in the image so users can optionally tag them. Another example is tools to automatically suggest tags for a tweet. What else can we do, and what general principles and tools can we provide?

Data curators. Data is no longer just in databases controlled by a DBA and curated by the IT department. Now, a wide variety of people are empowered to curate it. Crowdsourcing is one ap-

proach. A key challenge, then, is to obtain high-quality datasets from a process based on often-imperfect human curators. We need to build platforms that allow people to curate data easily and extend relevant applications to incorporate such curation. For these people-centric challenges, data provenance and explanation will be crucial, as will privacy and security.

Data consumers. People want to use messier data in complex ways, raising many challenges. In the enterprise, data consumers usually know how to ask SQL queries, over a structured database. Today's data consumers may not know how to formulate a query at all, for example, a journalist who wants to "find the average temperature of all cities with population over 100,000 in Florida" over a structured dataset. Enabling people to get such answers themselves requires new query interfaces, for example, based on multi-touch, not just console-based SQL. We need multimodal interfaces that combine visualization, querying, and navigation. When the query to ask is not clear, people need other ways to browse, explore, visualize, and mine the data, to make data consumption easier.

Online communities. People want to create, share, and manage data with other community members. They may want to collaboratively build community-specific knowledge bases, wikis, and tools to process data. For example, many researchers have created their own pages on Google Scholar, thereby contributing to this "community" knowledge base. Our challenge is to build tools to help communities produce usable data as well as to exploit, share, and mine it.

Community Challenges

In addition to research challenges, the database field faces many community issues. These include database education, data science, and research culture. Some of these are new, brought about by big data. Other issues, while not new, are exacerbated by big data and are becoming increasingly important.

Database education. The database technology taught in standard database courses today is increasingly disconnected from reality. It is rooted in the 1980s, when memory was small relative to database size, making I/O the bottleneck to most database operations, and when servers used relatively expensive single-core processors. Today, many databases fit in main memory, and many-core servers make parallelism and cache behavior critical to database performance. Moreover, although SQL DBMSs are still widely used, so are key-value stores, data stream processors, and MapReduce frameworks. It is time to rethink the database curriculum.

Data science. As we discussed earlier, big data has generated a rapidly growing demand for data scientists who can transform large volumes of data into actionable knowledge. Data scientists need skills not only in data management, but also in business intelligence, computer systems, mathematics, statistics, machine learning, and optimization. New cross-disciplinary programs are needed to provide this broad education. Successful research and educational efforts related to data science will require close collaboration with these other disciplines and with domain specialists. Big data presents computer science with an opportunity to influence the curricula of chemistry, earth sciences, sociology, physics, biology, and many other fields. The small computer science parts of those curricula could be grown and redirected to give data management and data science a more prominent role.

Research culture. Finally, there is much concern over the increased emphasis of citation counts instead of research impact. This discourages large systems projects, end-to-end tool building, and sharing of large datasets, since this work usually takes longer than solving point problems. Program committees that value technical depth on narrow topics over the potential for real impact are partly to blame. It is unclear how to change this culture. However, to pursue the big data agenda effectively, the field needs to return to a state where fewer publications per researcher per time unit is the norm, and where large systems projects, end-to-end tool sets, and data sharing are more highly valued.

Going Forward

This is an exciting time for database research. In the past it has been guided by, but also restricted by, the rigors of the enterprise and relational database systems. The rise of big data and the vision of a data-driven world present many exciting new research challenges related to processing big data; handling data diversity; exploiting new hardware, software, and cloud-based platforms; addressing the data life cycle, from creating data to analyzing and sharing it; and facing the diversity, roles, and number of people related to all aspects of data. It is also time to rethink approaches to education, involvement with data consumers, and our value system and its impact on how we evaluate, disseminate, and fund our research.

Acknowledgments. We thank the reviewers for invaluable suggestions. The Beckman meeting was supported by donations from the Professor Ram Kumar Memorial Foundation, Microsoft Corporation, and @WalmartLabs.

References

1. Abiteboul, S. et al. The Lowell database research self-assessment. *Commun. ACM 48*, 5 (May 2005), 111–118.
2. Agrawal, R. et al. The Claremont report on database research. *Commun. ACM 52*, 6 (June 2009), 56–65.
3. Apache Software Foundation. Apache Hadoop; http://hadoop.apache.org, accessed Sept. 12, 2014.
4. Apache Software Foundation. Apache Hive; http://hive.apache.org, accessed on Nov. 9, 2014.
5. Apache Software Foundation. Apache Pig; http://pig.apache.org, accessed on July 4, 2014.
6. Bernstein, P. et al. Future directions in DBMS research—The Laguna Beach participants. *ACM SIGMOD Record 18*, 1 (1989), 17–26.
7. Bernstein, P. et al. The Asilomar report on database research. *ACM SIGMOD Record 27*, 4 (1998), 74–80.
8. [C11] Cattell, R. Scalable SQL and NoSQL data stores. *SIGMOD Record 39*, 4 (2011), 12–27.
9. Dean, J. and Ghemawat, S. MapReduce: Simplified data processing on large clusters. *Commun. ACM 51*, 1 (2008), 107–113.
10. Silberschatz, A. et al. Strategic directions in database systems—breaking out of the box. *ACM Computing Surveys 28*, 4 (1996), 764–778.
11. Silberschatz, A., Stonebraker, M. and Ullman, J.D. Database systems: Achievements and opportunities. *Commun. ACM 34*, 10 (Oct. 1991), 110–120.
12. Silberschatz, A., Stonebraker, M. and Ullman, J.D. Database research: Achievements and opportunities into the 21st century. *ACM SIGMOD Record 25*, 1 (1996), 52–63.

The following authors served as editors of this article (the third author also served as corresponding author):

Philip A. Bernstein (philbe@microsoft.com) is a Distinguished Scientist at Microsoft Research, Redmond, WA.

Michael J. Carey (micarey@ics.uci.edu) is a professor in the Bren School of Information and Computer Sciences at the University of California, Irvine.

AnHai Doan (anhai@cs.wisc.edu) is a professor in the Department of Computer Science at the University of Wisconsin-Madison.