# The Seattle Report on Database Research

Daniel Abadi, Anastasia Ailamaki, David Andersen, Peter Bailis, Magdalena Balazinska, Philip Bernstein, Peter Boncz, Surajit Chaudhuri, Alvin Cheung, AnHai Doan, Luna Dong, Michael J. Franklin, Juliana Freire, Alon Halevy, Joseph M. Hellerstein, Stratos Idreos, Donald Kossmann, Tim Kraska, Sailesh Krishnamurthy, Volker Markl, Sergey Melnik, Tova Milo, C. Mohan, Thomas Neumann, Beng Chin Ooi, Fatma Ozcan, Jignesh Patel, Andrew Pavlo, Raluca Popa, Raghu Ramakrishnan, Christopher Ré, Michael Stonebraker and Dan Suciu

## ABSTRACT

Approximately every five years, a group of database researchers meet to do a self-assessment of our community, including reflections on our impact on the industry as well as challenges facing our research community. This report summarizes the discussion and conclusions of the 9th such meeting, held during October 9-10, 2018 in Seattle.

## 1. INTRODUCTION

From the inception of the field, academic database research has strongly influenced the state of the database industry and vice versa. The database community, both research and industry, has grown substantially over the years. The relational database market alone has revenue upwards of $50B. On the academic front, database researchers continue to be recognized with significant awards. With Michael Stonebraker's Turing Award in 2014, the community can now boast of four Turing Awards and three ACM Systems Software Awards.

The strong progress the database research community has made in recent years is clearly evident. Over the last decade, our research community pioneered the use of columnar storage, which is used in all commercial data analytic platforms, whether or not they are based on a relational engine. Database systems offered as cloud services are widely used and have witnessed explosive growth. Hybrid transactional/analytical processing (HTAP) systems are now an important segment of the industry. All data platforms have embraced SQL-style APIs as the predominant way to query and retrieve data. A new generation of data cleaning and data wrangling technology is being explored. Database researchers have played an important part in influencing the evolution of streaming data platforms as well as NoSQL systems.

Our achievements show that the state of our community is strong. Yet, in technology, the only constant is change. Today, we are living in a data-driven society where decisions are increasingly driven by the insights gathered from analysis of relevant data ("data is the new oil"). This societal transformation places us squarely in the center of technology disruptions. However, the fact that data is at the center of everything today also means that the field has grown in breadth and that new challenges have arisen. Indeed, just in the last five years, much has changed in industry and the research community. Technology trends are providing our community with an unprecedented opportunity to have an even bigger impact in today's data-driven world than ever before.

In the Fall of 2018, the authors of this report met in Seattle to identify and discuss research directions for the community that seem especially promising, considering the key developments that impact our field. There is a long tradition of such meetings in our community, held approximately every five years. The first such meeting was held in conjunction with VLDB 1988 [3] and the last one prior to Seattle took place in Irvine in 2013 [2].

This report summarizes the findings from the Seattle meeting of database researchers. We begin by reviewing key technology trends that impact our field. The central part of the report covers research themes that we believe are especially promising. We close by discussing steps the community can take to be more impactful beyond solving the technical research challenges.

## 2. WHAT HAS CHANGED IN THE LAST FIVE YEARS?

The transformation to data-driven decision making has been in progress for many years. In fact, the last report identified Big Data as our field's central challenge [2]. However, in the last five years, the transformation has accelerated well beyond our projections, in part due to technological breakthroughs in machine learning (ML) and artificial intelligence (AI). Deep neural networks (DNNs) have led to unprecedented progress in image analysis and natural language processing (NLP), among other disciplines. Reinforcement learning emerged as a powerful paradigm to complement traditional supervised learning. Recently, models such as BERT hold the promise of democratizing the use of natural language as an interaction model for tasks in any enterprise, not just Internet companies with a rich information corpus. The barrier to writing ML-based applications has been sharply lowered by widely available programming frameworks, such as TensorFlow and PyTorch, as well as new FPGA, GPU, and specialized hardware for use in private and public clouds. The database community has a lot to offer to ML users given our expertise in data discovery, versioning, cleaning and integration. These technologies are critical for a machine learning platform to derive insights from data. Execution of inference and training workflows can potentially benefit from query optimization techniques. The database community can also help shape how traditional SQL querying functionality is seamlessly integrated with machine learning. Moreover, with the increasing availability of usage data, ML can be leveraged to transform the database platform itself.

A related development has been the rise of data science as a discipline that combines elements of data cleaning, transformation, statistical analysis, data visualization, and machine learning techniques. Today's world of data science is quite different from the previous world of statistical tools such as SAS and SPSS, and from traditional data transformation tools in the enterprise data integration world. Notebooks have become by far the most popular interactive environment. Our expertise in declarative query languages can enrich the world of data science by making it more accessible to domain experts, especially those without traditional computer science background.

Our society has become increasingly concerned about the state of data governance. This is a difficult challenge as data moves within information systems as well as across organizational entities and national borders. Data governance requires that data owners adhere to data privacy and other constraints related to the movement of data. To meet this requirement, data provenance and metadata management technology are important ingredients. Data governance has also led to the rise of confidential cloud computing whose goal is to exploit cloud resources while keeping the data encrypted. Beyond data governance, another societal concern is ethical and fair use of data. This concern impacts all fields of computer science, but is especially important for data management, which must enforce such policies.

The last report observed that "Cloud Computing has become mainstream" [2], and indeed, usage of managed cloud data systems has grown tremendously in the last five years. As an alternative to provisioned resources, the industry now offers on-demand resources that provide extremely flexible elasticity, popularly referred to as *serverless*. For analytics, the industry has converged on a *data lake architecture*, which uses elastic compute services to analyze data in cloud storage "on-demand". The elastic compute could be jobs on a Big Data system such as Apache Spark, a traditional SQL data warehousing query engine, or an ML workflow. It operates on cloud storage with the network in-between. This architecture disaggregates compute and storage, so they can scale independently. These changes have profound implications on how we design future data systems.

Industrial Internet-of-Things (IoT), focusing on domains such as manufacturing, retail, and healthcare greatly accelerated in the last five years, aided by versatile connectivity, cloud data services, and data analytics infrastructure. Its requirements have further stress-tested our ability to do fast data ingestion and quickly discover insights with minimal delay for real-time scenarios such as monitoring. Their effectiveness also depends on efficient data processing at the edge, including data filtering, sampling, and aggregation.

Finally, there are significant changes in the hardware landscape. With the end of Dennard scaling, and the rise of compute-intensive workloads such as DNN, a new generation of powerful accelerators leveraging FPGAs, GPUs, and ASICs are being used. These technologies appear to be the only viable approaches today to train big models. The memory hierarchy continues to evolve with the ad-

vent of a new generation of SSD and low-latency NVRAM. Specialized interconnects as well as improvements in network bandwidth and latency continue to be remarkable. Beyond datacenters, the advent of 5G with ample bandwidth can reshape the workload characteristics of data platforms. These developments point to the need to leverage an increasingly heterogeneous hardware landscape as we rethink the architecture of the next generation of database engines.

## 3. RESEARCH CHALLENGES

While we have made progress in some of the key challenges articulated in the last report [2], many of the difficult questions remain relevant today. The changes described in the previous section present us with new scenarios that deserve our consideration as well. This report combines these two sets of research challenges, organized into four subsections. The first subsection addresses challenges in data science where our community can play a major role. The second focuses on the emerging societal concerns of data governance. The last two cover the cloud data services and the closely related topic of database engines. It should be noted that some of the challenges cut across multiple themes, e.g., machine learning.

### 3.1 Data Science

The NSF CISE Advisory Council[1] defines data science as a field that focuses on "*the processes and systems that enable the extraction of knowledge or insights from data in various forms, either structured or unstructured*" Over the past decade, it has emerged as a major interdisciplinary field and it will become even more important in the future.

Data science is used to drive important decisions in companies and discoveries in science. It is used for one-off decision making as well as for tracking Key Performance Indicators (KPIs) over time. From a technical standpoint, data science is about the pipeline from raw input data through data integration and wrangling, to data analysis, data visualization, and finally insights.

Through the history of database systems, users have extracted insights from their databases. They have used complex SQL queries, online analytical processing (OLAP), data mining techniques, and statistical software suites. The modern data scientist works in a different environment. Jupyter Notebooks are the new de-facto standard, and data scientists rely on a rich ecosystem of open-source libraries for sophisticated analysis, including the latest ML techniques. They also work with data lakes that hold structured and unstructured datasets with varying levels of data quality – a significant departure from carefully curated data warehouses. Furthermore, data science is fundamentally a multidisciplinary field with deep integration with an application domain, whether in science or industry. These characteristics have created new requirements for the database community to address, as discussed below.

*Data integration and wrangling*: Data scientists repeatedly say that data integration and data wrangling is 80-90% of their challenge. These are problems the database community has worked on for decades. Thus, it can bring a solid understanding of the core challenges and known solutions. In the past, we have focused much of our efforts on solving "point problems", e.g., algorithms for specific challenges such as entity resolution. Instead, we need to devote more efforts on the end-to-end data-to-insights pipeline, including understanding systems that go all the way from raw data to an end-user's desired outcome, such as a visualization of the answer to the user's question, or a prediction by a machine learning model.

*Data context and provenance*: Data scientists need to understand the quality of the results that they are getting from a data to insights pipeline. In traditional database applications, query results are assumed to be correct, complete, and fresh. The data is trusted because it was created by the same entity that consumed it. In modern applications, correctness, completeness, freshness, and trust cannot be taken for granted. Consumers need to know the degree to which these properties of their data hold and to reason about their impact. This requires understanding the context of the incoming data and the processes working on it. This is a classic data provenance problem, which involves tracking, integrating, and analyzing this metadata. Beyond explaining results, data provenance also enables reproducibility, which is key to data science, although it is especially difficult when data has a limited retention policy. This is an area where we need to focus our efforts.

*Data management in support of machine learning*: Data science pipelines require machine learning, including the latest techniques such as deep learning. The database community needs to embrace

[1] https://www.nsf.gov/cise/ac-data-science-report/CISEACDataScienceReport1.19.17.pdf

this new type of workload. In addition to developing efficient methods for executing such workloads (see Section 3.4), it needs to investigate declarative programming paradigms to specify and optimize all stages of machine learning pipelines (data discovery, data preparation, and model building). The management of models and machine learning experiments is an area where our community can provide rich support, including but not limited to model versioning. Data provenance is also important in machine learning, as it can help identify differences between test data and training data that cause models to lose accuracy.

*Fast Exploration*: To support exploratory analyses by data scientists, systems must provide interactive response times over Big Data, since high latency reduces the rate at which users make observations, draw generalizations, and generate hypotheses. More research is needed for at scale visualization and interactive query processing. Research is also needed in developing methods that make creating and debugging complex data science pipelines easier than writing code in an imperative language like Python. This work also requires a change in the database community conference culture, promoting user studies that assess the impact of novel technology on data scientists.

Modern data analysis and management, including data science, continues to move to public clouds where the data for analysis is drawn from data lakes. This change has significant implications, as discussed later in this report. Big Data, Data Science, and AI have also led to more diverse applications that often do not fit well with the relational model. Data management researchers should work on determining the right data models, query operators, storage schemes, and optimizers for emerging data-intensive applications. This requires participation in building end to end systems and experimentation with user applications.

Since the database research community has worked on many facets of data-to-insights pipeline, we are well positioned to have a big impact. However, we must collaborate with other disciplines so that our technological contributions are recognized and adopted widely. The time has come for our community to develop a data science agenda that builds on our strengths, attracts broad participation, and helps shape this emerging field.

## 3.2 Data Governance

The data management, and indeed the entire computer science community, has become socially aware. As technology continues to impact society in more profound ways, the database community must focus more attention on the societal impact of the technology they develop.

Today, end-users generate data that becomes input to many data-intensive applications. Some of it is about people: our homes have become "smart", with sensors located in doorbells, thermostats, and other appliances; virtual assistants have entered our living rooms; medical records are digitized; and social media is publicly available and widely popular. Data-intensive applications that use these data sources raise not only technical challenges but also those of privacy and ownership. Data producers have an economic and personal interest that the data is used only in certain ways. For instance, they might have licensed the use of their personal health records for medical research, but not for military applications. The European Union's General Data Protection Regulation (GDPR), which has gained wide adoption beyond Europe, is directly related to the issues of data privacy and data use. These topics of data governance are discussed below.

*Data use policy and data sharing*: In industry, data science pipelines are often complex and different sub-teams work on different steps of the pipeline: one team prepares the data; another builds models on the data, and yet another team accesses the data and models through interactive dashboards. Additionally, data science teams leverage multiple heterogeneous data sources in data lakes. The database community needs to develop tools that support collaborations around data, including labeling, annotating, exchanging, securing, discovering, and capturing provenance of data. Such collaborations and sharing must adhere to fine-grained access control and auditing requirements to ensure data is used by the right people for the right purpose. Data provenance technology is needed to support auditing at scale so that checks for legitimate usage can be implemented. Finally, as data volumes grow, we need to better techniques to compress data, move data to cold storage, and choose data to discard.

*Data Privacy*: As we continue to aggregate data, balancing the need for data privacy with analytical usage of such data for decision support has emerged as a key challenge. Cryptographic techniques as well as differential privacy have emerged

as a foundation for much of the privacy work, including in our community. However, it is still unclear in what way differential privacy may be embedded in the database platforms effectively without restricting the query surface significantly. Collaborations across organizations are also subject to privacy constraints and require techniques such as Multi-Party Computation.

*Ethical data science*: Machine learning models can contribute to bias and discrimination. Activities around surfacing and countering those problems have gained traction in research and practice. The bias often comes from the input data itself. Sometimes, it comes from insufficiently representative data used to train models. Our community can apply expertise in data quality and data integration to help address this problem. Responsible data management has emerged as a new research direction where the data management community has much to contribute. A related challenge is identifying data designed to misinform, e.g., on social media platforms. Addressing the above challenge requires inferring intent, as well as collaboration with the NLP, computer vision, and other communities.

## 3.3 Cloud Services

The movement of workloads to the cloud has led to explosive growth for cloud database services, which in turn has led to substantial innovation, experimentation, as well as new research challenges.

*Challenges of new consumption models*: The simplest consumption model is Infrastructure-as-a-Service (IaaS). This model is very flexible, but users must handle all operational management of the database system themselves. An emerging trend for such services is to exploit innovations such as "spot pricing" in the underlying IaaS services to optimize costs for non-critical workloads. In contrast to IaaS, managed services, offered either by first-party cloud providers or third-party multi-cloud vendors, sharply reduce operational complexity but provide less flexibility. When managed services were introduced, users paid for them by a provisioned capacity model. Alternative consumption models have now emerged: usage-based pricing as well as hybrid models that support on-demand event-driven auto-scaling of compute and storage. As we continue to move away from pre-provisioned resources to on-demand elastic infrastructure, including serverless, new challenges for state management arise. What will be the best way to offer serverless database services with the pay-as-you-go on-demand model? Such event-driven on-the-fly creation of data services will have a significant impact on the architecture of the query or the storage engines. Two other key difficulties for users of cloud data services are the absence of SLAs for cloud data services that go beyond availability and the lack of transparency on how auto-scaling and other choices affect the cost.

*Challenges of cloud architecture*: The cloud architecture raises unique opportunities and challenges for innovative database system design:

- *Disaggregation*: An important characteristic of cloud architectures is the use of very large pools of commodity hardware that are subject to hardware and software failures at scale. To handle such failures, modern cloud databases are increasingly decoupling storage and compute for high availability, scalability, and durability. For example, all distributed data lake platforms have been or are being rearchitected with compute and storage services decoupled. Disaggregation is key to enabling elastic compute, but its ability to meet response time requirements critically depends on caching effectively, which is challenging due to the multiple levels of memory hierarchy. It also depends on supporting some minimal compute within the storage service that can sharply reduce data movement. (See also Section 3.4)

- *Multi-tenancy:* Unlike traditional environments where resources are scarce and carefully provisioned per workload, cloud-based infrastructure offers an opportunity to rethink databases in a world with an abundance of resources that can be pooled together for a set of workloads. In such an environment, it is critical to support multi-tenancy to control costs and utilization. This requires mechanisms to respond swiftly and alleviate resource pressure as demand causes local spikes. Telemetry can be used to predict usage and take proactive measures. Over longer time frames, there are challenges of capacity management. The range of required innovation here spans reimagining database systems as composite single-tenant and multi-tenant microservices, creating and operating predictive models, developing mechanisms for agile response to resource demands, reorganizing resources among active tenants dynamically without impacting active application workloads and ensuring tenants are isolated from noisy neighbor tenants.

- *Hybrid cloud*: In an ideal world, on-premise data platforms would seamlessly draw upon compute and storage resources available in the cloud "on-demand". There is a pressing need to identify architectural approaches that make it possible for on-premise data infrastructure and cloud systems to take advantage of each other instead of relying on "cloud only" or "on-premise only". As enterprises split their data processing across on-premise systems and the cloud, they need a single control plane for the entire data estate.

- *Edge and cloud*: IoT has resulted in a skyrocketing number of computing devices connected to the cloud, in some cases only intermittently. The limited capabilities of these devices, characteristics of their connectivity (e.g., limited bandwidth for offshore devices, ample bandwidth for 5G-connected devices), and their data profiles will lead to new optimization challenges for distributed data processing and analytics.

*Leveraging uniqueness of SaaS*: Software-as-a-Service (SaaS) applications need to be multi-tenant. But in contrast to ad-hoc multi-tenancy, each tenant has approximately or exactly the same database schema (but no shared data) and the same application code. One way to support multi-tenant SaaS applications is to have all tenants share one database instance with the logic to support multi-tenancy pushed into the application stack. While this is simple to support from a database platform perspective, this approach makes customization (e.g., schema evolution), query optimization, and insulation from a noisy neighbor harder. The other extreme approach is to spawn a separate database instance for each tenant. While flexible, this approach is not cost-effective as it fails to take advantage of commonality among tenants. Yet another approach is to pack tenants into shards with large tenants placed in shards of their own. Such packing of tenants into shards is nontrivial. Moreover, security considerations may also constrain the specific architecture that is chosen. Thus, there is a need to think carefully about tradeoffs between design alternatives in architecting SaaS and about functionality that needs to be supported at the cloud database infrastructure vs. implemented in the application stack.

*Multi data center issues*: Cloud applications that operate across multiple data centers, potentially geographically distant from each other, remain a key challenge for both analytics and active-active online transaction processing (OLTP) workloads (see details in Section 3.4). Some countries have data sovereignty laws that make it illegal to move their citizens' data to another country's data center. More work is needed to understand how these considerations impact data center replication and high availability.

*Auto-tuning*: Cloud databases need to support a diverse set of time-varying multi-tenant workloads, and no one configuration tuning works well universally. Furthermore, the vastly expanded user base of cloud databases lacks expert DBAs. Studies of cloud workloads indicate that many cloud database applications do not use best practices for configuration settings, schema design, or data access code. Thus, while auto-tuning has always been important, for cloud databases it is of critical importance. Fortunately, cloud systems' telemetry logs are plentiful and present a great opportunity to improve the auto-tuning functionality of the database systems. Machine learning may be helpful here as well.

*Confidential cloud computing*: Enterprises are concerned about security and privacy of their data when it moves into a public cloud. This has led to the rise of confidential cloud computing, which makes data visible only to the enterprise, so no security lapse in the cloud infrastructure compromises its privacy. The challenge is to enable this functionality with an acceptable loss of performance. While there has been progress in this area, a comprehensive approach to confidential cloud computing is yet to emerge and thus this remains a fertile research area.

*Opportunity for data sharing*: The cloud offers a unique opportunity for flexible data sharing. More importantly, we need to define architectures for data sharing idioms. In its most stringent form, data sharing can be viewed as a variant of the multi-party computation (related to confidential cloud computing described above). In its simplest form, it is the ability to leverage public data sets along with private data sets. We should explore other idioms of data sharing between these extremes. A related challenge is a scalable flexible search for data sets and providing provenance and other related metadata that are crucial for data sharing.

Minimizing lock-in for cloud data services and facilitating interoperability across multi-clouds will benefit all users. Today, each public cloud is a

"walled garden", created using rich supporting tools around data services. While important, there has been little effort in facilitating multi-cloud either in the industry or in the research community.

### 3.4 Database Engines

As mentioned earlier in the report, the last two decades have witnessed significant changes that have impacted the architecture of data platforms. One change affecting the core database engines is the rise of scalable distributed "document-stores", which support key-value look up and horizontal scaling. Another is the evolution of the Hadoop ecosystem to a more efficient Spark ecosystem for extract-transform-load jobs (ETL) and support for relational execution over such a runtime by leveraging query processing techniques from database engines. New memory-optimized data structures, compilation, and code-generation have significantly enhanced performance of traditional database engines. Main-memory database techniques have become established in both industry and research, often as part of HTAP systems. Another key achievement of the field are highly scalable streaming systems that are widely used. All data analytics engines have now implemented column-oriented storage. The cloud has reinvigorated work in geo-distributed replication, and the industry has made significant progress on that front. As mentioned earlier, the need for elastic computation in cloud platforms has led to rearchitecting database engines for disaggregated storage and compute.

We now discuss the key themes related to the evolution of database engines.

*Heterogeneous computation*: We see a clear trend towards heterogeneous computation with the death of Dennard scaling and the advent of new accelerators introduced to offload compute. GPUs and FPGAs are available today, with the software stack for GPUs much better developed than that for FPGAs. Likewise, we see increasing deployments of RDMA. The memory and storage hierarchy is also more heterogeneous than ever before. The advent of high-speed SSDs has already had significant performance impact and altered the traditional trade-offs between in-memory systems and disk-based database engines. Engines with the new generation of SSDs are destined to erode some of the key benefits of in-memory systems. Furthermore, NVRAM is finally becoming generally available, which might have significant impact on database engines due to their support for persistence and low latency. Embracing this new normal world of heterogeneous hardware and re-architecting the database engine accordingly will be one of the most important tasks for the community. We also need to explore what would be an ideal hardware-software co-design that will best support database engines. For example, a co-design could help with asynchronously doing bulk lookup of objects for queries. Thus, we expect database architects to have an active agenda to take advantage of disaggregation, recent and upcoming hardware innovations, and to explore hardware-software co-design.

*Data lakes and modern data warehousing applications*: The needs of traditional data warehousing applications have expanded. They need to consume data from a variety of data sources. They need to transform the data and perform complex analyses more quickly. These new requirements have a profound impact on the design of core database engines that support them. The community is in the middle of a transition from classical data warehouses to a data-lake-oriented architecture for analytics. Although popularized in the public cloud due to the wide availability of scalable low-cost blob storage, a data lake architecture is equally applicable for on-premise systems. Instead of a traditional setting where data is ingested into an OLTP store and then swept into a data warehouse through an ETL process, perhaps powered by a Big Data framework such as Spark, the data lake is a flexible repository that can ingest a variety of data objects. Subsequently, a variety of compute engines can operate on the data, to curate it or execute complex SQL queries, and store the results back in the data lake or send it to other operational systems. Thus, data lakes exemplify a disaggregated architecture. One unique challenge of data lakes is *scalable data discovery*. Therefore, *data profiling*, which provides a statistical characterization of data, is of utmost importance in data lakes. Data profiling is challenging for data lakes, as profiling must have low latency despite having to provide a statistical summary of very large, heterogeneous, and possibly semi-structured data sets. Other challenges include finding all data relevant to a task quickly, e.g., identifying data that is joinable with other relevant data sets after suitable transformations.

*Leveraging approximation*: As the volume of data continues to explode, we must seek techniques that reduce latency or increase throughput of query processing. For example, leveraging approximation for *fast progressive visualization* of answers to queries

over data lakes can help exploratory data analysis to unlock insights in data. Data sketches have already gone mainstream and are classic examples of effective approximations. Sampling is another tool that can be used to reduce the cost of query processing. However, the support for sampling in today's Big Data systems is quite limited and does not cater to the richness of query languages such as SQL. Our community has done much foundational work in approximate query processing, but we need a better way to expose it in a programmer-friendly manner with clear semantics.

*Distributed transactions*: Data management systems are increasingly being distributed across multiple machines both within a single region and across multiple geographic regions. This has renewed interest in industry and academia on the challenges of processing distributed transactions. The increased complexity and variability of failure scenarios, combined with increased communication latency and performance variability in distributed architectures has resulted in a wide array of tradeoffs between consistency, isolation level, availability, latency, throughput under contention, elasticity, and scalability. There is an ongoing debate between two schools of thought: (1) Distributed transactions are hard to process at scale with high throughput and availability and low latency without giving up some traditional transactional guarantees. Therefore, consistency and isolation guarantees are reduced at the expense of increased developer complexity. (2) The complexity of implementing a bug-free application is extremely high unless the system guarantees strong consistency and isolation. Therefore, the system should offer the best throughput, availability, and low-latency service it can, without sacrificing correctness guarantees. This debate will likely not be fully resolved anytime soon, and industry will offer systems consistent with each school of thought. However, it is critical that application bugs and limitations in practice that result from weaker system guarantees be better identified and quantified, and tools be built to help application developers using both types of system achieve their correctness and performance goals.

*Leveraging machine learning*: Recent advances in ML have inspired our community to reflect on how some of hard data engine problems could use ML to advance the state of the art. The most obvious such problems are in *auto tuning*. For example, we can systematically replace "magic numbers" in database systems with data-driven learning models and use them to auto-tune system configurations. ML also provides new hope for progress in query optimization, which has seen only minor improvements in the last two decades. Although in principle almost any component can potentially be improved with ML, answers to some key questions are prerequisites for success, such as availability of training data, a well-thought software engineering pipeline to support an ML component (debuggability is notoriously hard), and availability of the guard-rails so that when test data or test queries deviate from the training data and training queries, the system degrades gracefully.

*Support for machine learning in database engines*: As we briefly discussed in Section 3.1, modern data management workloads include ML. This adds an important, new requirement for database engines. We must immediately address the challenge of efficiently supporting "in-database" ML. Today, this is achieved by leveraging the traditional extensibility mechanisms of databases. However, as DNN models become more popular and bigger, supporting efficient inferencing and training will require database engines to leverage heterogeneous hardware and support popular ML programming frameworks. This evolution is still in its early stage. Database engine architects need to work with architects responsible for building ML infrastructure using FPGAs, GPUs and specialized ASICs.

*Benchmarking*: Over the years, benchmarks tremendously helped move forward the database industry and the database research community. The traditional benchmarks (e.g., TPC- E, TPC-DS, TPC-H) that the database community has developed are good, but they do not capture the full breadth and depth of our field. We need to reaffirm our commitment to invest in benchmarking for the new application scenarios and database engine architectures. For example, without development of appropriate benchmarking and data sets, a fair comparison between traditional database architectures and ML-inspired architectural modifications to database systems will be impossible. The community needs new benchmarks that capture the needs of modern workloads, in particular with respect to the velocity and variety dimensions of Big Data, e.g., streaming scenarios, data with skew, workloads with common data transformations, and processing of new types of data such as videos. While some benchmarks are emerging in this space, much more remains to be done. A closely related issue is that of poor practice of performance evaluation in publications. The

selection of workloads, databases, and parameters often lacks rigor. Moreover, only a simplistic aggregate metric (average) is typically reported, instead of providing important additional metrics such as the variance.

*SQL Standard*: While the SQL Standard has been a major benefit to the ecosystem, SQL implementations in different data systems still differ in semantics. Our community must continue to push towards making SQL a true standard. At the same time, SQL may not be enough for supporting data science applications and ML workloads. Therefore, we need to investigate systems that combine relational algebra and linear algebra in a richer query paradigm, potentially as extensions to SQL.

Two "holy grails" should continue to stay on our agenda. First, we must always explore any novel ideas to reduce the impedance mismatch between application development and writing database queries. Second, we must continue to find ways to make database systems less rigid (e.g., flexible schema evolution) without significantly sacrificing their performance.

## 4. COMMUNITY

The database community is in a healthy state with stable numbers of submissions to our conferences. Our flagship conferences are well attended by academics and engineers from the industry. Nevertheless, the community continues to have great opportunities to improve, as discussed below.

*End-to-end solutions in the hands of users*: To increase its impact, the database research community should put more weight on developing (or participating in the building of) full-fledged systems, as well as use these systems and tools to help real users. Incorporating the algorithmic innovations from our community into working systems will greatly increase their impact and reach. Moreover, by interacting with real users, the community will increase its impact, visibility, and connection with today's rapidly changing data management challenges.

*Open source and Cloud Services*: To achieve high impact, our community should develop tools that are easy to adopt. This will be the case if they are part of existing, popular ecosystems of open-source tools or are available as easy-to-use cloud services. Specifically, an important way through which we can have impact is when we rally around large-scale open-source systems. Such systems accelerate innovations because new algorithmic ideas can be readily added to them. They also become mature tools, with large communities of developers that can more easily support real users. Recent examples of such systems that either came out (or, included significant input from) the database community include Apache Spark, Apache Flink, and Apache Kafka.

*Data science software ecosystem*: The database community must do a better job integrating database research with the data science ecosystem (e.g., Jupyter notebooks, Python). Database techniques for data integration, data cleaning, data processing, and data visualization should be easily called from Python scripts. The community can also develop more elegant APIs that scale more naturally and better integrate relational and linear algebras. These tools need to work well at any scale, not just for the biggest problems. Users should be able to try the tools at small scale on their laptops or in the cloud and should not have to make significant changes to their code as their data and computational needs grow.

*Community innovation*: The database community has always been innovating its approach to running conferences, such as having multiple conference deadlines per year, changing how papers are assigned and reviewed, pioneering the evaluation of the reproducibility of accepted papers, expanding the conference programs to include tutorials. Going forward, it is important for the community to remain innovative and to continue to improve the paper selection processes. Several factors have surfaced in recent years. First, database researchers now have tremendous opportunities for ambitious research projects, large centers, and exciting collaborations with industry. As a result, many researchers are too busy to participate in program committees as often as they would like, even for our top conferences, because they are simply stretched too thin. We need to find a solution to this problem because we want the input of all our community members in the reviewing process. Second, paper reproducibility continues to be a challenge. The community has experimented with many techniques to encourage researchers to make their work open-source and reproducible, but this continues to be an uphill battle. The community needs to continue to explore innovative ideas to remedy this problem. Third, the current review process and common understanding among reviewers honors algorithmic research contributions as "syntactically correct papers", with a clear definition of a baseline that an

algorithm improves upon, whereas some members of the community do not sufficiently value other contributions, such as papers describing innovative systems, applications, experiments and analyses, or user studies. One way to improve the situation is to more strongly emphasize "potential for impact" in the reviewing process.

*Impact on university campuses*: The advent of data science and the excitement around data science education, including data science minors, majors, master's degrees, and specializations within existing majors, are great opportunities for the database community to broadly influence education on our university campuses. Data science is multidisciplinary. It includes components from machine learning, human computer interaction, statistics, ethics, law, data visualization, application domains, and of course data management. Many aspects of data management are very relevant to data science, as discussed in Section 3.1 of this report. Students from all fields of study (not only computer science) need to learn the technology our community has developed over the years to support the data to insights pipeline. Therefore, database experts are natural collaborators for teaching data science and devising data science curricula. Database faculty should thus be engaged in discussions that define the data science curriculum on their respective campuses.

## 5. LOOKING FORWARD

It is impossible to capture fully the exciting discussions we have had in our meeting in Fall 2018. This report summarizes some of the key recommendations and reflections from that meeting. A downloadable copy of this report as well as some supplementary materials used in the meeting can be found on the event website [1].

Beyond the summary recommendations in this report, database researchers should be attentive to technology and application trends. Much has already changed since our Fall 2018 meeting. Every new mechanism that has emerged offers a potential opportunity to enhance data management capabilities (e.g., blockchain, quantum computing) and every new scenario is a potential application area where data management might help (e.g., self-driving cars, fake news).

As the database community continues its strong history of impact on research and industry, we are fortunate to have many exciting research directions around data science, machine learning, data governance, as well as new architectures for cloud systems and data engines. We need to focus on building more impactful open-source software systems and cloud services based on our research, and better integrate with the existing data science stack and tools. In our conferences, we must revisit how scholarship is evaluated and recognized so that we are set up for maximal impact.

## 6. ACKNOWLEDGMENTS

The Seattle database meeting was supported financially by donations from Google, Megagon Labs, and Microsoft Corporation.

## 7. REFERENCES

[1] The Database Research Self-Assessment Meeting 2018. http://seattle-dbresearch-assessment.cs.washington.edu, 2018.

[2] D. Abadi, R. Agrawal, A. Ailamaki, M. Balazinska, P. A. Bernstein, M. J. Carey, S. Chaudhuri, J. Dean, A. Doan, M. J. Franklin, J. Gehrke, L. M. Haas, A. Y. Halevy, J. M. Hellerstein, Y. E. Ioannidis, H. V. Jagadish, D. Kossmann, S. Madden, S. Mehrotra, T. Milo, J. F. Naughton, R. Ramakrishnan, V. Markl, C. Olston, B. C. Ooi, C. Ré, D. Suciu, M. Stonebraker, T. Walter, and J. Widom. The Beckman Report on Database Research. *CACM*, 59(2):92–99, 2016.

[3] P. Bernstein, U. Dayal, D. DeWitt, D. Gawlick, J. Gray, M. Jarke, B. Lindsay, P. Lockemann, D. Maier, E. Neuhold, A. Reuter, L. Rowe, H.-J. Schek, J. Schmidt, M. Schrefl, and M. Stonebraker. Future Directions in DBMS Research – The Laguna Beach Participants. *ACM SIGMOD Record*, 18:17–26, 03 1989.