O'REILLY®

Compliments of SingleStore

The Rise of Operational Analytics

How the NewSQL Revolution Enables Real-Time Decision Making and Continuous Insight

Scott Haines

REPORT

SingleStore

The Database of Now

SingleStore is purpose-built to deliver "Analytics with an SLA"

Speed
SingleStore delivers the fastest event-to-insight performance with our breakthrough lock-free architecture

Scale
SingleStore offers unbeatable scale, delivering consistent performance under high ingest and concurrency

SQL
SingleStore plugs right into standard SQL, BI, and ETL products as well as modern tools like Kafka and Spark

Try it FREE today at singlestore.com

The Rise of Operational Analytics

How the NewSQL Revolution Enables Real-Time Decision Making and Continuous Insight

Scott Haines

Beijing • Boston • Farnham • Sebastopol • Tokyo
O'REILLY®

The Rise of Operational Analytics
by Scott Haines

Published by O’Reilly Media, Inc., 1005 Gravenstein Highway North, Sebastopol, CA 95472.

O’Reilly books may be purchased for educational, business, or sales promotional use. Online editions are also available for most titles (*http://oreilly.com*). For more information, contact our corporate/institutional sales department: 800-998-9938 or *corporate@oreilly.com*.

Acquistions Editor: Jessica Haberman
Developmental Editor: Nicole Tache
Production Editor: Kristen Brown
Copyeditor: Octal Publishing, LLC
Interior Designer: David Futato
Cover Designer: Karen Montgomery
Illustrator: Rebecca Demarest

October 2019: First Edition

Revision History for the First Edition
2019-10-02: First Release

978-1-492-07366-6

[LSI]

Table of Contents

Foreword

Much has been said about the critical importance of data today. Data has been described as everything from "the new oil" and "the new currency," to "a powerful weapon" in quiet wars—whether for customers, or between countries. But it's not data by itself that is the real source of value. How we leverage data to make decisions, to respond to events, and to deliver the ideal product or experience is where value is created.

We call this new business capability **Operational Analytics—analysis and instantly available insights woven throughout our operations**, feeding dashboards and alerts, applications and automation, wherever in the organization it's needed. In a world where the way in which we use data creates competitive advantage, a number of new truths emerge.

Firstly, **every company must become more insight-driven**. This means that, instead of a handful of specialists querying data in a closed system, successful companies have hundreds or thousands of employees hammering analytics systems every day in order to make better informed decisions. In addition, we see the rise of automated systems, which are also querying diverse data sets to feed their algorithms and actions.

Second, **it is no longer acceptable to wait for the next quarter, month, week, or even day** to get the insight needed to make a business decision, or provide feedback. Customers, competitors, and markets are moving and changing faster than ever before, and the ability to learn, adapt, and evolve is the most important capability of a modern business. Companies that can access streaming, real-time data about their business have a significant edge over the competition.

Lastly, **legacy data management and analysis systems were not designed to support the scale of today's data-driven business**. Just as we saw with digital transformation, this new data revolution is requiring us to rethink our processes, organizational structures, skill sets, and the tools we use to support the future of our businesses. The ability to effectively ingest, transform, analyze, expose, and transact with data is the very foundation of this new approach to business.

This is what drives SingleStore every day, and why we worked with O'Reilly to support the writing of this book. SingleStore was founded to help businesses leverage data in order to reach their full potential, and its focus continues to be helping you redefine the limits of what's possible with data.

— Peter Guagenti
Chief Marketing Ofcer,
SingleStore October 2019

CHAPTER 1

How Data Drives Innovation

The Evolution of Information

For most of history, access to information was limited to the elite—the wealthy, and scholars and philosophers they sponsored—who had access to a small number of precious texts, laboriously copied by hand. Beginning with Gutenberg's printing press, around 1440, people of all backgrounds and social strata gained access to knowledge through the printed word.

A new breakthrough came in the form of the telegraph, arriving in the 1860s. Better-informed decisions could be made thanks to electrical transmission of data. By the late 1800s, Alexander Graham Bell had his first patent on the telephone. Soon, humanity had a medium for real-time, interactive communication.

By the mid-20th century, mainframe computers arrived, speeding up common tasks like mathematics and text processing. Computer-to-computer communication demanded a more robust alternative to long-distance telephony, leading to the birth of the internet. This marked the start of the digital Information Age. As had been the case with manuscripts, access was initially limited to an elite group, but then reached mass audiences.

Today, companies find themselves with global networks of always-on customers. Each customer demands access to products and services on their own schedule and wherever they happen to be. There is no longer any "close of business." Engagement with customers, suppliers, partners, and others has never been as important as it is now.

Data, especially event-based and behavior-based information, is tied to huge gains in the performance of the data-savvy enterprise. The ability to generate insight from customer and user behavior has never been as important as it is at this precise moment.

The Data-Driven Company

Businesses are now in the midst of a new technological revolution in which the *speed of decision making* and *insight into mission-critical operations* are defining an entire new category of real-time, data-driven companies.

These businesses understand that the speed with which they can access, share, and build upon well-defined datasets, across business units, through unified data platforms is one of the most important differentiators that they can take advantage of to propel innovation and creativity across their industry. Reports that are produced weekly, even daily, are no longer fast enough to capture and define new emerging trends that form within business operations, across financial markets, from the operation of myriad devices, and across global supply chains.

Fast access to all of your company's data, live and historical, has now become the critical game changer. It is up to system architects and data engineers to build and maintain data platforms that can ingest, define, and sanitize massive amounts of continuous events and metrics data, combining it with in-depth historical data to feed the minute-to-minute operational needs of the business.

These needs are met through tools such as truly up-to-date business dashboards, decisioning tools operating on the latest and greatest information, customer-facing applications that reflect both the organization's accumulated knowledge and the current environment (both virtual and real) in which the user is operating, and more.

As a recent Gartner report states, "by 2022, more than half of major new business systems will be incorporating continuous intelligence that uses real-time context data to improve decisions." However, when asked, only around "12% of companies are currently starting to integrate streaming data into their analytics requirements." In other words, the industry knows the switch from batch input to real-time data ingest is a big deal in terms of competitive advantage.

The companies who build out their next-generation data platforms now will have a leg up on their competition.

Key Advantages of the Data-Driven Company

Companies that have already embraced this shift can be seen using their data to drive some really incredible services, with features that would have been out of reach just a few years back. Here are a few examples of the capabilities and trends that have emerged over the past few years:

- Media companies that sell ads in real time, up-to-the-second auctions, maximizing the business value of users' website visits, drive-time radio listening, and television watching.
- Real-time logistics systems controlling and optimizing the coordination of human workforces, as seen with customer-facing tools such as Amazon Prime and operational tools such as human capital management software.
- System-wide monitoring of large-scale physical device networks such as next-generation smart water management for agriculture as well as sensor networks in, for example, natural gas pipelines (Internet of Things [IoT]).
- Banking systems that detect cyber- and real-world security issues, protect customers' funds, and improve portfolio management, even moving credit card fraud detection from an overnight process to fighting fraud "on the swipe," as the transaction is actually being made.
- Almost instantaneous feedback loops and state management for connected homes (IoT), including light networks, garage doors, even coffee makers and barbeque grills.
- Smart navigation systems that use machine learning and predictive analytics to route around problems in real time, as seen with Waze and other connected mapping solutions.
- Feed-forward systems that consume and aggregate user event data to provide better time resolution in cases like customer support and crisis support (as seen with the Crisis Text Line).

CHAPTER 2

The Rise of Operational Analytics

Decision Making at the Speed of Business

Companies today need to make informed decisions regarding the minute-to-minute behavior of their business. Gone are the days of waiting for weeks between progress reports, which were generated laboriously and then shipped off to recipients for semi-manual analysis. Even nightly reports delivered the next business day are often out-of-date when considering what is currently happening within daily business operations. Bottom line, the faster you can make an informed decision, the faster you can provide value to your internal and external stakeholders.

Over the past decade or two, the shift from batch to real-time data processing has been a pivotal trend. The result? New platforms and frameworks that bring near-real-time processing speeds while preserving the advantages of tried and true frameworks.

The shift has been increasing in momentum due to the lower cost of running cloud-based resources (such as servers and databases), the widespread availability of well-maintained and well-documented open source frameworks, as well as demand by both businesses and consumers to move faster and smarter.

Companies today are indeed making faster decisions, using the capabilities described in this report. By doing so, they are outcompeting their peers. Let's take a look at how this transformation began.

The Emergence of Operational Analytics

Tools that enable streaming data have helped to redefine how organizations process big data with respect to ingestion, transformation, and integration across various additional data platform systems and services. They have also helped lay the groundwork to prepare the world for fast analytics processing on a mix of historical and fresh, near-real-time data, which is becoming known as *operational analytics*.

Operational analytics can help drive daily business operations and increase profits, all the while creating competitive advantage for the organization by presenting a view of the most up-to-date information pertaining to day-to-day business functions. We can further break down operational analytics by looking at the requirements of what we will refer to as *can't wait* and *won't wait* data.

"Can't Wait" Data for Decision Making

"Can't wait" data refers to data used internally for predictive analytics, reporting, and dashboards. It has become necessary to enable critical decision making across the entire business. Rising in popularity has been the widespread use of real-time dashboards, powered by in-memory caches containing the most up-to-date data available to the company. These in-memory caches achieve near-real-time status by directly ingesting event data using stream processing technologies.

These dashboards provide stakeholders with the means to view and join data from across the organization, as well as mechanisms to support just-in-time (JIT) aggregation and native support of statistical correlation. In addition, data visualizations and interactive dashboard environments act as conduits to the data, enabling nonprogrammers to interact with data in a similar fashion as their more technical fellow team members. These dashboards can also provide a means to generate monitoring of business goals that are at risk and other critical subsets of the company's data. These monitors can be set up to trigger when specific thresholds are crossed or when anomalies arise outside the normal behavior of a given metric stream.

Dispatch of these notifications to internal or external stakeholders—as in the case of fraud or distributed denial-of-service attacks

(DDoS)—can be done in many ways, but two of the more popular services used are Slack and PagerDuty.

Slack is a very popular messaging platform used to communicate with members over channels. Distributed notifications among channels within a company or organization are enabled through application webhooks, making it easy to integrate with server-side applications.

PagerDuty is a system commonly used by engineering teams across the globe to notify and alert when things go wrong within critical business systems. Within site reliability or first line of defense Network Operations Center, or "NOC" teams, hooking into PagerDuty is just an API call away.

The underlying behavior of the queries made through shared operational analytics systems must be taken into account when optimizing for "can't wait" decision making capabilities. Consider the business intelligence (BI) use case. BI systems are generally connected to data warehouses and have been built up to generate specialized reports tailored to specific business statistics such as daily/monthly revenue per active user, customer churn statistics based off of daily/monthly active unique users, and myriad other SQL-based queries across large amounts of data.

These reports tend to be run in periodic schedules like daily/last24, weekly/week over week, monthly/month-over-month. The results of these aggregations can be ingested back into operational analytics databases, providing point-in-time reference of specific metric aggregations that can be used to provide almost-instant support for critical decision making. These specific rollups provide the historic trend data, whereas the overlay of the near-real-time, current event data can be used to optimize for statistical distance problems like Euclidean Distance or alternative correlation techniques.

Identifying new and emerging sectors, such as when logistics-as-a-service companies like Doordash, Lyft, and Uber are looking to enter new markets or ramp up in underrepresented delivery areas, typically requires longer-running queries, or a series of many procedural queries to cross-correlate the staggering metadata of hundreds of thousands of rides and deliveries.

The process of exploring the massive amounts of event data typically employs Bayesian statistics and machine learning in order to identify outliers or to group related data into hierarchical clusters. This data can be sliced and diced in almost any fashion, and can help spot trends like an increase in average wait time per delivery or canceled rides per hour/day. Again, feeding these kinds of historical metrics back into the core operational analytics system can help open up new avenues of exploration for the critical decision makers.

"Won't Wait" Data for Customer Experiences

The data demands needed to meet customer requirements are increasing at exponential rates. We refer to these demands as "won't wait" data, where a customer will simply stop using a system that fails to meet their needs and expectations. Instead, the customer will go to a rival offering a similar service, find another way to meet the same requirement, or simply get distracted and do something else entirely.

Consider two well-known companies mentioned just earlier, DoorDash and Uber. Customers who place an order for food or who request a ride to a destination such as an airport want to be informed of exactly what is going on. Time is of the utmost importance, and being able to get updates in real time of the progress of a ride or delivery, or the ability to smartly predict where drivers will be in demand, are just two examples of the importance of up-to-the-moment data in customer satisfaction.

These logistics systems will make recommendations in real time, and also store the event stream of a given order or ride in an archival database or filesystem to apply data mining and machine learning techniques in a more controlled, offline environment. This analytical work is used to improve customer interactions and satisfaction in the future.

Being able to tie these customer behavior metrics back into the business operational analytics systems enables decision makers to forecast future needs for drivers or increase marketing/sales pushes to underrepresented delivery areas to drive usage.

Additionally, customers of Software-as-a-Service (SaaS) applications expect to have near-real-time dashboards of the services they have built experiences and companies around. Consider a business making a decision regarding which vendor to select when moving forward with a new conferencing solution. This decision will determine the foundation upon which the company meets across distances, engaging with customers, partners, suppliers, and among employees. Competing vendors may have good SLAs and promise 99.x% uptimes, but the ability for a vendor to provide metrics and live dashboards can make the difference between landing the deal or not.

For both "can't wait" operational requirements and "won't wait" customer experiences, companies are often spurred to take action when increasing volumes of incoming data, or increasing demands by more users and apps to connect and consume data—that is, increasing concurrency—causes Service-Level Agreements (SLAs) to be broken. Sometimes, companies simply take the minimum action needed to restore SLA compliance; in other cases, they rethink what's possible and reorganize systems to achieve performance that far exceeds previous SLA levels, thereby gaining benefits in operational efficiencies, customer satisfaction, usage of their services, and profits.

Single Source of Truth

For organizations that have embraced the need for machine learning and artificial intelligence (AI), one of the biggest stumbling blocks—after hiring qualified experts—is the general availability of quality structured data that can be consumed by the data science organization. The availability of data drives the capabilities of the data science teams. Access to both labeled and unlabeled data enables core machine learning capabilities like supervised and unsupervised learning across datasets.

Having the ability to query a single-source-of-truth data store that can return complete and trustworthy data, bridging both historic as well as real-time data, is emerging as a trend across the industry. Projects like MLFlow and Kubeflow have established patterns for organizations to follow in order to get the most out of cross-functional teams working within the machine learning/AI space. It has become imperative to optimize the performance of these organizations by providing tools catered toward simplifying the life cycle of machine learning/AI workflows, which in turn yields faster results and more productive teams.

Operational analytics will drive the future decision making of business leaders by enabling them to make informed decisions that encapsulate all available data, both historic and real-time, in a unified way. This allows the business to use domain knowledge, insights, and intuition as a lens to fuel the future of growth for the business in a data-driven way.

Next, we take a look at common problems organizations face while getting their operational analytics systems to market.

CHAPTER 3

Challenges with Data Platform Infrastructure

The Trouble with Building Up a Data Platform

Across all industries, growth in data generation is exploding inexorably, gaining in size, extent, and speed like an avalanche in motion. With the shift from 4G to 5G networks, telecommunications companies have opened the door to gigabit (>1 GBps) bandwidth speeds. Considering that 4G LTE bandwidth tops out at up to 300 MBps, the increase in data volume and speeds alone will enable mobile connected services to produce and consume entire orders of magnitude more data globally.

This includes Internet of Things (IoT) sensor data, higher-resolution real-time event streams, and usage statistics from mobile applications, as well as native 4k video streams. With the new capability, we can expect more devices to be connected as well.

Scalability across all components within the data platform is of vital importance to enable the elastic growth necessary to handle the processing requirements for all that lies on the road ahead. Realizing and harnessing the value of the data produced across the business will drive more autonomous decision making as well as providing a necessary, unbiased "voice" when making critical business decisions. Lastly, as companies are trending toward a "store everything" attitude with respect to their data, architects must always be focused on providing key infrastructure to help reduce friction and enable holistic solutions to be used across the company.

Selecting the correct initial platform capabilities and deliverables provides a stable base upon which many more core foundational platform components can be built. Like a foundation, keystone, or cornerstone, these building blocks will be used to support future additions and enhancements within the platform. A transparent strategic vision and clear blueprint of the road ahead can help guide the team around the quick-win style shortcuts that inevitably lead to critical design flaws in the fundamental architecture of the data platform.

In the next section, we take a look at the key common components of the modern data platform and understand why they must exist in order to support near-real-time operational analytics systems.

Database Types

Databases are core containers that hold and store a company's most valuable assets. The data that your business generates comes in many shapes and sizes, and the way in which that data is queried should be considered when deciding what capabilities are necessary for a given database technology.

Given the general proliferation of databases over the years, it's useful to look at the main categories of databases in order to understand what capabilities define a given database technology and to understand what challenges might need to be addressed.

OLTP Databases

The *online transaction processing* (OLTP) database has been around for a while and is probably the best known and understood, conceptually, in the industry. The makers of these databases pride themselves on their reliability and accuracy when it comes to transactions, or sequential mutations—updates—to data stored within them.

These databases adhere strictly to what is known as ACID compliance. ACID is an acronym that stands for *Atomic*, *Consistent*, *Isolated*, and *Durable*.

Within the OLTP database there are tables comprised of many rows of structured data, as depicted in Figure 3-1. Each row of data can be further broken down into one or more columns (or fields), which can be thought of as pointers to atomic data of a specific type.

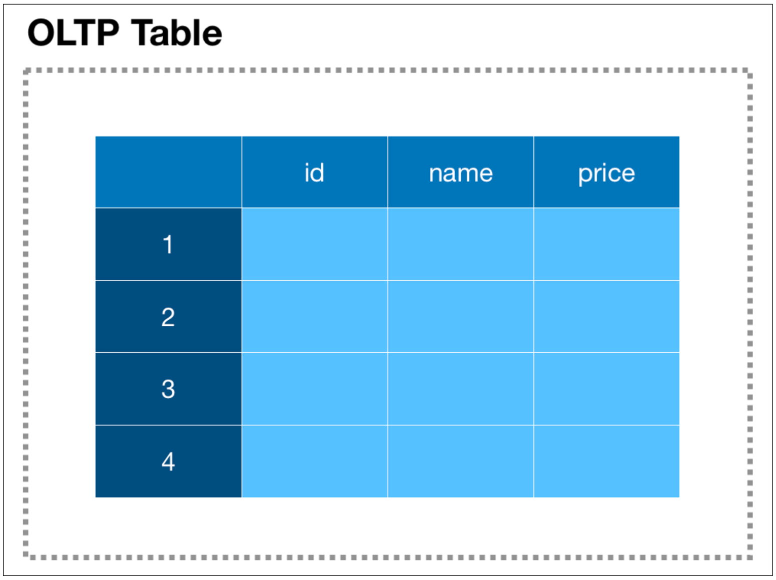

Figure 3-1. Database rows and columns

Given that these systems support multiple connected user sessions, accuracy and consistency of the underlying data contained within these databases is critical. This capability is made possible by the transactional architecture within the database engine. If one user is to manipulate the value of a field within a row of data—for example, in the case of an ATM withdrawal from a shared bank account—a subsequent withdrawal from a different location, by a different user from the same account, should reflect the prior withdrawal as well as the current withdrawal, regardless of the current state of the data currently cached or available at that location.

These transactions are isolated events that will only succeed after the transaction has been completed and the underlying value recorded. This consistency through sequential, isolated events maintains a durable record, or one that exists even if the database suddenly goes offline, or experiences connectivity hiccups.

These databases are traditionally relational in nature, and exist under the umbrella term *relational database management systems*

(RDBMS). This includes popular RDBMS flavors such as MySQL and PostgreSQL. An RDBMS allows data from one or more tables to be joined to generate new views of data, while also adhering to strict user access and permission policies known as *grants*.

These capabilities make OLTP the first choice for most organizations as a general-purpose platform for accessing data, given all that is provided out of the box. The core language used to query these systems is *structured query language* (SQL), which is supported by most OLTP and also most Online Analytical Processing (OLAP) systems (see the upcoming discussion). SQL is the most common language used to interact with most databases.

Even though there are many upsides to traditional OLTP databases, the downside is that scalability comes at a much higher cost, given that the architectures of these systems maintain ACID compliance at the cost of not being able to scale out across industry-standard, low-cost machines. Their architecture is referred to as a scale-up or shared-everything architecture because to scale beyond the system's current memory and available disk size, larger hardware must be provisioned. Scaling out—horizontal scalability—is not supported.

OLAP Databases

The OLAP database is optimized to provide fast responses to both single-dimensional analytical queries, such as queries within a specific data table, and multidimensional analytical queries, which require data to be combined across multiple tables. These queries analyze data across a search space from many different vantage points, referred to as dimensions, and provide analytical query support, which can be categorized under the umbrella of *rollup*, *drill-down*, and *slice-and-dice* aggregation.

These capabilities provide a means for an analyst to efficiently explore the data, in aggregate, across the subset of a bounded search space within what are called *cubes* or *hypercubes*. For example, the approach to typical data mining takes a top-down approach starting with a rollup, which is a high-level aggregate view of all data within a search space. To uncover interesting patterns and enable further analysis of the data, the next step is to drill down into subsets within the dimensional space.

This is achieved in an opposite approach as the rollup, in that data is aggregated across subsets of specific dimensionality, as in the example of drilling into performance characteristics from a regional level, down to the state level and finally across a city, or subdivision level like area code or zip code. As a means to investigate more finely, an analyst will slice and dice across any level, from the initial rollup to some level of drill-down, to look at how patterns emerge across multiple dimensional filters.

An example of this would be an analyst producing a business impact report based on the results of a specific targeted marketing campaign, to understand the impact across critical markets when compared to normal nontargeted areas. In a now infamous story of transaction analysis for items purchased together, a retailer found that there was a high correlation between late-night purchases of diapers and beer. This data was used to put beer closer to the diaper aisle, and sales of both beer and diapers continued to skyrocket.

OLAP engines, and newer search engines, can provide critically necessary analytics support via traditional Business Intelligence (BI) systems. OLAP engines enable complex queries to be executed for purposes such as data mining, trend analysis, forecasting, cohort analysis, and many more. Enabling faster analytics and report generation can also help boost revenue across the company by enabling sales and marketing organizations to move quicker and follow more intelligently selected leads and other data-driven opportunities.

Although there are many upsides to OLAP, this style of query provides scaling challenges in different ways than OLTP. Instead of supporting *narrow and isolated* transactions, OLAP queries *operate widely* across multiple fact tables to support on-the-fly aggregations to answer focused analytical queries.

These queries can become performance bottlenecks, especially when run on top of existing OLTP databases, due to the nature of what is required to support transactions (ACID) within a shared-everything architecture with a single active node. To address the scalability concerns that arise when running queries on top of OLTP systems, companies transition OLAP processing onto read-only databases. Removing the write/transaction capabilities allows the database to utilize RAM (memory) and CPU more efficiently because the concerns of maintaining atomic mutable state go away. In other words, transactions can then run on one dedicated system, and queries on another, speeding up both. As use cases continue to grow in size and complexity, they eventually run separate hardware and software stacks in order to support continued scalability.

NoSQL Databases

The portrait of the modern data platform wouldn't be complete without mention of *NoSQL*. NoSQL databases became popular in the 2000s as a means to tackle the lack of horizontal scalability within the available RDBMS systems of that era. NoSQL databases provide key/value, document, as well as graph-style data stores, among other types, across distributed keyspaces. These keys act as pointers to memory or disk space allocated to store arbitrary blobs of data within the data nodes of the system.

Although these data stores can grow horizontally and maintain high availability, this comes at a cost to consistency and, more generally, a lack of schema—more or less rigid structures for arranging data. Values associated with a given key can contain any kind of value. As a result, companies had to either enact strict internal policies as to what can go in a key (or other) field, or deal with additional schema inference when analytics are run against the data. Ensuring that data could be extracted in a predefined format was a complication to these schema-less data stores.

Cassandra, Redis, and MongoDB are some of the more well-known NoSQL databases that emerged during the NoSQL movement that are still popular and being maintained today. Each of these databases constructed a query paradigm that feels SQL-like, while not adhering strictly to the specifications for ANSI SQL.

Cassandra has CQL, which uses keyspace indexes and table definitions to add search capabilities to the underlying key/value store. Redis created RedisSearch to provide secondary index capabilities and full-text search to the data stored within its in-memory data structure store. Lastly, MongoDB offers a map-reduce style search approach over its document store and have defined their own SQL-like nomenclature, which is different enough that they provide a conceptual mapping from SQL to MongoDB parlance to assist people new to their database offering.

NoSQL stores can provide high availability and horizontal scalability over lower-cost hardware, as well as distributed fault-tolerance, at the cost of consistency. They can, however, provide a very good solution to fetching append-only or write-once data that can be used to record a great deal of data quickly and efficiently while shifting much of the burden of making the data useful to the querying process.

Besides the three types of general-purpose databases described here —OLTP, OLAP, and NoSQL—there is a fourth type, *NewSQL*, which we describe in Chapter 4. We then compare all four types of databases to each other in Chapter 5.

Special-Purpose Data Stores

The aforementioned major database types can each be used for a wide variety of purposes, and they have given rise to variants as well. Here you'll find the major categories of database-like systems powered by the database types above.

Data Warehouses

The data warehouse was envisioned as a means to help mitigate the myriad challenges that arise from querying data across many disparate sources of siloed data, produced across the company by many different teams, and to provide unified data to power BI environments for the purpose of making informed business decisions from a "single source of truth" data store.

The challenges of working with data tend to cluster more generally around naming and context, data formats (DateTime versus Date versus Long), inconsistent temporal information (UTC versus PST or system.time), missing values and nullability, data lineage, and general data availability. To address the challenges of standardization, the data warehouse emerged to become the final resting place for clean and normalized source-of-truth, business-centric data.

To get data into the data warehouse, data from one or more sources is extracted, transformed, and loaded into the data warehouse through an externally scheduled or on-demand batch job, which is known as an extract, transform, and load (ETL) job. The data imported to the data warehouse is historic in nature, due to the time it takes to batch input data, run transactions, and run the ETL process. Data in a data warehouse can be made more up-to-date with the use of data lineage rules and strict data SLAs.

Data Lakes

The data lake is a general purpose, highly scalable, low-cost storage tier for raw data. If the data warehouse is a source of purposeful structured data that has been preprocessed and loaded to answer *specific* business-centric questions, the data lake is the staging area for data that *might* have importance in the future.

Given that businesses are moving to a "store everything" mentality across most of the data they produce, it is essential to have a storage tier that can scale to meet the storage needs of tomorrow while also providing benefits today.

The challenges of storing raw data, however, can manifest when the time comes to make use of and provide purpose to these deep pools of raw potential. The challenge is that over the months and years that data is sitting and waiting, changes to the data being produced and a lack of records as to what exactly was being recorded at various points can make older data unusable. This can dramatically increase the lead time to make use of the data or cause years of data to be thrown away due to corruption of data over time.

So even though the data lake is conceptually a storage area for raw, underpurposed data, there is still a need to provide a moderate level of effort in order to ensure that data corruption doesn't occur along the way, and that the business can make practical use of the data when the time comes to put it into action.

Distributed File Systems

To provide a horizontally scalable file system that can accommodate the needs of the data warehouse or the data lake, technologies like Apache Hadoop have been utilized, due to the overall performance that comes from an atomic, highly available, rack-aware, fault-tolerant distributed filesystem. Hadoop handles massive data growth through a scale-out architecture that can utilize commodity hardware or specialized hardware, hard-disk drives (HDDs) or solid-state drives (SSDs). The underlying distributed filesystem that ships with Hadoop is called HDFS, which stands for the Hadoop Distributed File System. HDFS can scale out to handle billions of files across a multipetabyte distributed filesystem.

Managing data has its ups and downs, and with large volumes of data, it can be difficult to scale without dedicated teams of administrators who are focused on the health and well-being of these distributed data stores. So, to handle the growing pains of scaling almost infinitely, cloud vendors like Amazon Web Services (AWS), Microsoft Azure, and Google Cloud Platform (GCP) all have provided an offering that "acts like" HDFS and promises a 99.9x% uptime guarantee.

Although there are definitely many upsides to farming out these services, the downside can actually be hidden. In the case of Amazon Simple Storage Service (Amazon S3), the filesystem is eventually consistent. This comes at a performance cost when compared to native HDFS, which can do atomic writes, especially when you consider the role these file systems play within your data platform.

Data Pipelines

Lastly, we arrive at the data pipeline. The data pipeline can be used to connect the different types of just-described databases and data stores to each other, or to connect any of them to ingest sources, such as sensors or data entry systems.

This component is strategically essential to connecting all of the various components within the data platform. You can think of these pipelines in the same way as the assembly line is to car manufacturing.

In the beginning, they start out with some raw material, like event or sensor data, generated from within one of many services running within a company's product offerings. This data is then ingested through a service gateway or API that act as a means of initially validating, authenticating, and eventually enqueuing data into one or many ingress (entry) points within the larger data pipeline network.

Once the data has entered the pipeline, it is free to then move through a predefined series of steps which typically include sanitizing or cleaning, normalizing or fitting to a schema, and, potentially, joining with other data sources. At one or many egress (exit) points, this data can be used to satisfy the data needs of another downstream system. This process of specific transformations is referred to as *data lineage*, and these multiphasic processing steps are enabled through the use of the data pipeline, but are not actors within a physical data pipeline.

This can come across as a confusing concept, but the data pipeline is a name given to the actual processing of data, in a controlled and repeatable way, whereas the sources of data can reside across any data store within the data platform.

Recently there has been a move toward streaming data pipelines that can be tapped into at specific predefined junction points, enabling new data processing capabilities that more effectively reuse their upstream transformations. One of the key technologies that has led to the proliferation of streaming data pipelines is Apache Kafka.

Kafka allows data processing pipelines to be constructed by connecting applications through a series of event-specific topics, which are associated with a specific type of structured or semi-structured data (such as JSON, Avro, or Protocol Buffers) that encapsulates a specific event or message type.

Consider the assembly line. Raw materials and components arrive at one or more ingress points. Each station works on one specific job, and at the egress point, a car pops out. In the event streams use case, each station along the proverbial assembly line represents an application, one which consumes data from a specific data topic, processes the stream of event data, and feeds the results along to the next application via another topic.

This pattern scales out to generate exponentially expanding data networks, enabling organizations of all sizes to operate in more agile ways by connecting various data-processing applications to these streaming data networks. For example, this enables new paths to fill the data lake as well as more optimized ETL flows, which can develop from batch to streaming through the use of available streaming data. These streams ultimately enable lower end-to-end latency, which in turn yields faster ingest times to the data warehouse and connected operational analytics systems.

Spark and the Shift from Batch to Streaming

One of the many projects that helped to lead the transformation to faster and smarter decision making is Apache Spark, created by Matei Zaharia in 2009 and donated to the Apache Foundation in 2013. Spark started out as a batch-processing framework that could run alongside, or on top of, the Hadoop ecosystem.

The focus of the project was to easily distribute MapReduce-style distributed compute jobs across horizontally scaling infrastructure, in a resilient way, that was at the same time highly available and fault tolerant. This new approach to distribution of work was made possible by generating transformative lineage-aware datasets that are called Resilient Distributed Data (RDD). The use of RDD and the Spark compute framework led to speed and performance gains across the platform that reach the level of orders of magnitude, when compared to similar workloads run in a standard MapReduce framework such as Hadoop.

Today, Spark is one of the most popular big data frameworks of all time and has added support to handle streaming data, machine learning at scale, as well as a unified API for graph search. Spark is now considered part of the standard toolkit for machine learning and AI projects.

Kafka Takes Streaming Further

Around the same time that Spark was gaining popularity, LinkedIn was making huge strides in the distributed producer/consumer (pub/sub) space with a project called Kafka, officially open-sourced through the Apache Foundation in 2011.

Kafka can be utilized as a data platform Swiss Army knife. This one platform can solve dozens of complex data engineering problems, but is best known as a general stream-processing platform for event data. Apache Kafka has taken the ideas implemented in Spark and generalized them in such a way that, today, Kafka and Spark are often used together.

At its core, Apache Kafka simplifies how data is written to a streaming resource, through what is known as a Kafka *topic*. Topics are the top-level resource. You can think of them as scalable collections of specific time-series events.

Topics can be configured to have one or more *partitions* of data that achieve high availability through the notion of *replicas*. These replicas are then strategically stored across a cluster of machines to allow for a "zero point of failure" architecture.

Data that resides on servers, called *brokers* in Kafka, can be written to by *producers* who write data into a given topic, and read out by *consumers* of the topic, as demonstrated in Figure 3-2. Scaling out a topic is as simple as adding more partitions, which can increase both read and write throughput as brokers suffer from less contention in reading and writing to their associated topics.

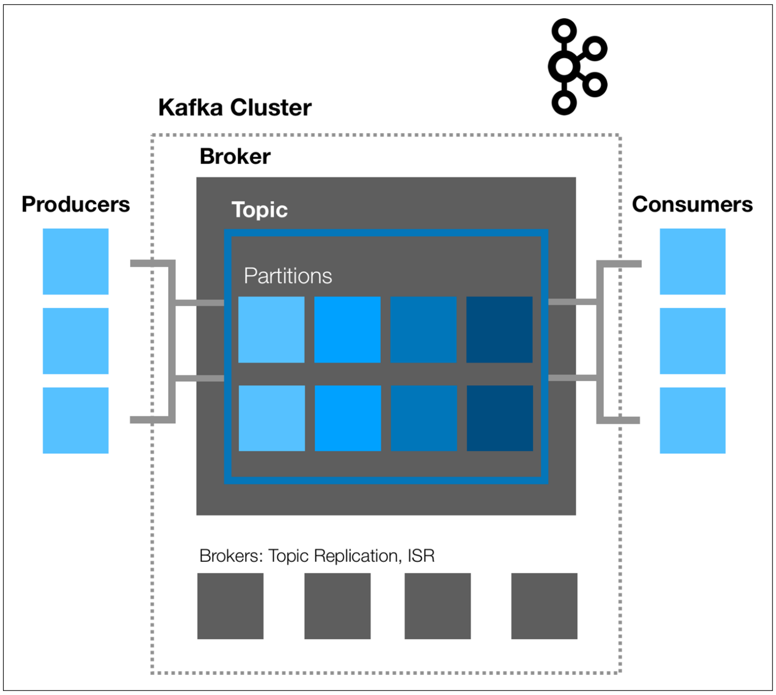

Figure 3-2. Kafka architecture with an example of a Producer and Consumer model

Kafka usage across the data industry has grown immensely for many reasons. For example, it is the perfect solution for time-series data, given that insertion into a topic partition will maintain explicit synchronous order in a first-in, first-out (FIFO) paradigm. This is critically important for maintaining a clear and accurate stream of events to the downstream consumer of a specific topic. Kafka can be used to broadcast data as it becomes available, without causing site-wide reliability issues.

As is often the case with innovation, initial ideas are made better through collaboration and widespread accessibility. Apache Spark and Kafka, both open source projects, benefited from widespread participation across the tech community. These tools have helped to provide the building blocks for continuous-streaming systems. They serve as a solid foundation for further progress and innovation. In the next chapter we look at how these concepts helped ready the path for operational analytics.

Achieving Near-Real-Time Capabilities for Data Transfer

Achieving near-real-time capabilities requires a different approach to the use of the different database types, data stores, and data transfer capabilities described earlier. In an operationally oriented system, a Kafka data pipeline brings data from transaction systems to an operational database. Analytics run against the operational database. It might even be possible to integrate transactions into the operational data store, greatly speeding and simplifying data flows. Noncritical data can be stored in a data lake and used for record keeping, data science experimentation, and other, less-critical purposes.

Companies are spending billions of dollars annually moving data around their systems, and as SpiceWorks pointed out in its 2020 state of IT report, "89% of companies are going to maintain or increase their IT budget," with "56% of larger-sized companies (5,000 employees or more) seeing their budgets increasing."

These increases are targeted toward replacing older IT systems, investing in new initiatives, handling new regulations like the European Union's General Data Protection Regulation (GDPR) and other compliance certifications, and growing teams in key IT sectors. These budgets are also used to purchase SaaS solutions to pave the way into new, emerging markets, without the long lead times traditionally associated with building up new infrastructure in house.

Depending on the size and capabilities of the data platform, teams can easily take three to five years to build out all the requisite components, build trust through uptime and stability, and provide the self-service capabilities that help to simplify access and adoption of these critical data platform components. Appropriate selection of self-service SaaS solutions can help the company focus the budget on more critical projects.

A simpler, speedier approach to data processing and data transfer can ultimately save money and make it easier to integrate externally provided solutions, such as SaaS solutions and public cloud platforms, with a company's in-house software and systems.

CHAPTER 4

NewSQL and Operational Analytics

The NewSQL Revolution

Timing is everything. Considering the progress the technology and open source community had made up until this point in time, the foundation was laid for the next evolutionary step in SQL processing. As we've seen, we started with single node online transaction processor (OLTP) database engines, then moved into hybrid OLTP/online analytical processing (OLAP) engines, then onto dedicated NoSQL systems, which could be used in conjunction with one another to solve problems across many domains.

To provide the base for each set of additional innovations, across cloud computing and distributed systems, new design patterns and architectures were envisioned. These patterns helped to open up an entire new set of capabilities only dreamed of prior to today. These capabilities allow for massive horizontally scaling, reliable and durable distributed systems, as well as frameworks for processing streams of data as we saw with both Kafka and Spark.

These steps paved the way for NewSQL and operational analytics. NewSQL combines the best of traditional relational systems with the inherent scalability formerly found only in NoSQL; it has been endorsed by industry analysts, such as Gartner and Forrester, as we describe in a moment. We look further into these topics now.

What Exactly Is NewSQL?

NewSQL databases represent the marriage of the ACID–based transactional consistency and reliability of OLTP databases to the high availability, horizontal scalability, and fault tolerance found in NoSQL databases. NewSQL databases have been redesigned from the ground up to support cloud-native architecture while balancing the trade-offs between consistency, availability, and fault tolerance (network partition-tolerance) that are required to run highly elastic, distributed systems within the cloud.

These new databases establish a holistic ecosystem that supports business needs across both the OLTP (RDBMS) requirements as well as the Business Intelligence (BI)–style OLAP workflows, all within a single hybrid system. (Gartner has coined the term HTAP, which stands for Hybrid Transactional Analytical Processing, for this new marriage.) It is through this merger of cross-functional capabilities that the stage has been set for the revolutionary operational database.

Gartner has described HTAP as creating new opportunities for dramatic innovation by business. Forrester calls this style of processing *translytical*, combining transactional and analytical processing in a single database, and it also sees bright prospects ahead.

A Brief Note on NewSQL and Database Theory

The CAP theorem governs the choices that can be made in database design. It basically states that you can have only two of the three desired characteristics of a database at any given time: consistency, availability, and partition tolerance. For NewSQL databases, partition tolerance is taken as a given because scalability requires it.

NewSQL databases can be balanced to meet operational needs of many different styles of workloads, which are split between consistency and partition tolerance (CP) or availability and partition tolerance (AP), while acting in some cases as a CP/AP load balancer to achieve standard SLAs within a highly elastic environment.

The Operational Database and Operational Analytics

Given the separation of concerns (SoC) between OLTP and OLAP queries, these hybrid databases must provide flexible distributed query processing that can continue to power mission-critical BI search queries while at the same time ingesting a near-continuous torrent of events and metrics data as well as transactions from across operational, edge, and mobile systems. They must provide an accurate view of not only up-to-date data, but also be capable of analyzing the equivalent of the entire data warehouse worth of data on demand, in order to achieve the very fast query response times required to answer the ever-growing demands of the business.

Key Features of the Operational Database

The operational database needs to effectively scale to handle enormous, unbounded datasets that support simultaneous ingestion of continuous data while at the same time supporting uninterrupted complex queries from hundreds to thousands of concurrent users (human users sending SQL queries, BI programs, and apps, including newer apps powered by machine learning and AI). The defining features of these databases are as follows.

Zero Downtime

In the modern age of cloud computing and SaaS, there is a zero-tolerance policy to service interruptions and any kind of downtime. Planned or unplanned maintenance can cost your company thousands, even millions of dollars, as seen with Amazon's PrimeDay outage in 2018. Your operational database must be highly available, fault-tolerant, and self-healing.

Shared-Nothing Scale-Out

Operational databases achieve high availability, fault tolerance, and consistency by incorporating a shared-nothing, scale-out architecture. This is achieved when data exists only where it needs to, and resides only in more than one place in order to maintain fault-tolerance. This is achieved within an active primary shard/passive multireplica shards-style data network. The use of gossip-style protocols allows each node within the distributed database to act as either a query processing node, a search relay node, or a coordination node for data ingestion and primary and recovery replication. The analogy of gossip here takes its roots from *evolutionary psychology*, in which gossip helped improve cooperation among people by directly affecting their social status, thereby keeping people honest. By allowing distributed components to "gossip," there are no secrets in the system and therefore no single points of failure.

The ability of each node within the stack to act on a query removes the need for a single master node and enables low end-to-end latency by running queries at the source of the data across a locally distributed subcluster. This architecture also enables strong durability for transactional data as explicit primary data nodes receive an update, write the change, and broadcast the change before a success state is returned.

Balance Consistency and Availability

Operational databases favor consistency over availability while at the same time achieving high availability and fault tolerance through the shared-nothing, scale-out architecture. NewSQL databases like the one delivered by SingleStore enable tuning between consistency and availability to achieve the correct balance between CP and AP to handle the various needs of different business use cases.

High-Performance Data Ingestion

Loading data into the operational database should be high performance while adhering to strict data availability SLAs and also achieving very low (subsecond) end-to-end refresh latencies—that is, with respect to the refresh of underlying collections of data backing the database's logical tables. Ingestion of data should also be simplified by enabling continuous loading of data through the use of existing external and database-specific data pipelines.

Streaming data sources like Apache Kafka, or the continuous ingestion of file streams from Amazon S3, Microsoft Azure Blob, or HDFS, which are common sources of data stored within a business' data warehouse or data lake, should be standard capabilities that ship with an operational database. Change Data Capture (CDC) also comes into play here, as operational databases many times are expected to live alongside systems of record. Ingestion of new data and refresh of the views of data within the database should also not affect or interfere with system-wide query performance.

Fast Query Response Time

Operational databases need to elastically scale up to handle a massive number of concurrent queries made by hundreds and even thousands of connected users, with the same speed and reliability of traditional OLTP databases, while also supporting the BI/OLAP join, rollup, drilldown, and aggregate workloads. Query response time is not measured in minutes, and in many cases not even seconds, but in milliseconds. Powering critical decision making requires subsecond response times on your business's most up-to-date data in order to drive insights and clearly highlight emerging trends.

Some databases, like SingleStore, use a mixture of highly optimized columnar data stored on disk, which achieves a high level of com-pression, while also achieving extremely fast aggregate queries as well as in-memory (RAM) rowstores. This supports fast point quer-ies, aggregations, and full row/columnar indices across the data stored in their hybrid database. This achieves the best of both OLTP and OLAP while still supporting full ANSI SQL as well as all analyti-cal SQL language features expected to support the myriad BI use cases.

Security to Stake Your Company On

Having a fast, reliable database also means having very strict security policies. Data access-level security policies protect the data within a shared database environment across all connected users. Grant-style permissions, like those that became popular for securing OLTP databases, provide the basic table-view-level access rules within the database. It has also become popular to include column-level encryption/decryption policies controlled by user-defined functions (UDFs) or one-way hashes to ensure personal information or restricted data doesn't leak out to just any user. Enabling data owners to view critical information while also adhering to extremely strict security policies, as seen with European Union's GDPR and Health Insurance Portability and Accountability Act (HIPAA) compliance, is a must. Additionally, end-to-end transport-level encryption and encryption at rest are a must have for enterprise security.

Use Cases

Now that we understand what NewSQL is and how the operational database enables unified transactional and hybrid analytics workflows, we next explore how the integration of these new databases can be transformative when mixed into large enterprise architectures. We will see how complexity can be reduced while opening the door to more consistent, predictable workflows.

Given that Kafka is the leading open source solution for event streaming, and increasingly the platform of choice for data-driven interoperability, it is also the most supported streaming connector across SaaS offerings and open source projects. Apache Spark works natively with Kafka and provides an additional connectivity layer to most database technologies available on the market through the use of plug-ins, whereas Kafka's stream processing paradigm enables exactly-once processing across Spark applications in many use cases, and is a native component of SingleStore's Pipelines ingestion archi-tecture as well as Apache Druid and Apache Pinot.

Now let's dive deeper into two use cases across two separate industries to see how pivoting toward operational analytics has led to revolutionary changes across not just data architecture, but across businesses as a whole.

Ecommerce

Fanatics, a global leader in the licensed sports merchandise game, with more than $2 billion in annual revenue, had a complex data architecture, with different data sources for different internal users. It wanted to streamline its analytics and data architecture in order to provide a more holistic environment that could be used by data scientists, data engineers, executives, and nontechnical stakeholders alike. And the ecommerce platform had to seamlessly adapt and react to both planned and unplanned events—as in the case of championship game results, player trades, and injuries.

Moving to an Event-Driven Architecture

Fanatics had recently made the decision to go through a larger, more substantially complex architectural transformation that saw it move to a purely event-driven streaming data architecture, from an older monolithic and deeply siloed system of separate applications. This transformative move happened over a three-year period and introduced the "FanFlow" Kafka-based event bus—a flexible, event-based data delivery conduit that provides streams of data to power core data systems (see Figure 4-1).

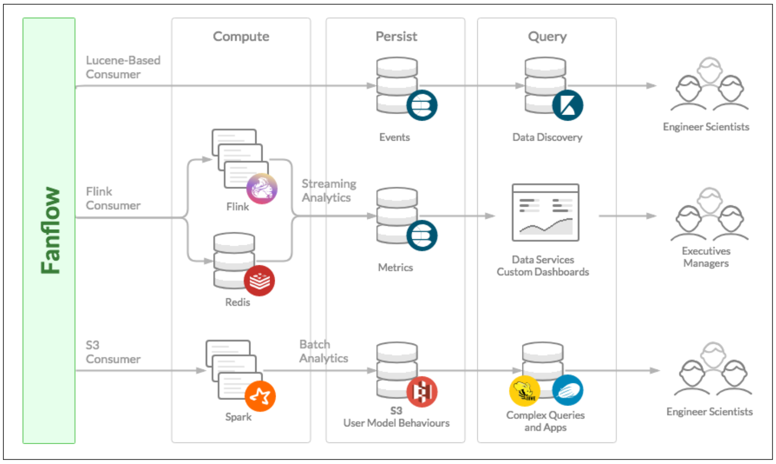

Figure 4-1. Before: The previous Fanflow analytics architecture used different tools for different audiences

These events encapsulate the state changes within an order life cycle, such as additions to cart, checkout, fulfillment, and delivery, as well as user behavior interactions on the site, type-ahead suggestions and point-of-sale events.

This reinvestment was a monumental step in the right direction, but Fanatics still had some larger issues to resolve on its journey to a truly unified analytics platform. These problems existed in the form of three separate systems, one which was a Lucene-based search index that saved raw events, a second which was an Apache Flink consumer with a Redis State machine that provided streaming analytics and metrics to their executives via custom dashboards, and, lastly, an Amazon S3 consumer that fed event streams into Fanatics' Spark clusters to model user behavior and support ad hoc queries from its Hive data stores, which powered an Apache Zeppelin notebook environment.

This split between tools and functionality for internal stakeholders manifested problems due to not sharing a single source-of-truth data store, and views of data for the executives would not be synchronized with the Lucene-based stores, or the data view from the Hive/Zeppelin dashboards.

An additional problem also occurred as the company's Lucene-based indexer couldn't keep up with huge peak traffic spikes, and the trickle-down effect created issues that led to trouble managing data-level SLAs across its system.

Shifting to a Unified Operational Architecture. An operational database replaced the Lucene-based indexers, and Spark and Flink jobs were converted to SQL-based processing jobs (Figure 4-2), which allowed for a more consistent, predictable development life cycle and more stable SLAs. Now, internal stakeholders can all operate off of the same underlying up-to-date, near-real-time data, while not sacrificing long development cycles maintaining and managing multiple clusters running on many different technology stacks. This was a game changer and a huge win for Fanatics.

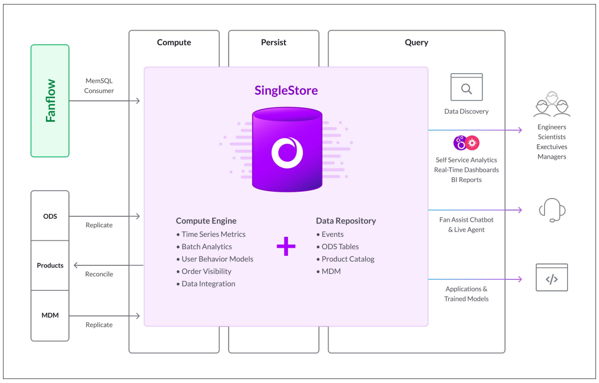

Figure 4-2. After: Fanflow feeds into robust analytics capabilities powered by SingleStore

Telecommunications

The telephone dates back to the late 1800s, and yet, over almost a century and a half later, the industry is continuing to make leaps and bounds with respect to the overarching technology. Telephony now enables global connectivity across all continents in the form of voice, video, and data. However, for many telecommunications companies, the industry-wide move to cloud-native stacks can seem out of reach, given that huge budgets were previously poured into custom hardware, special-purpose microprocessors, and proprietary software as a means to edge out the competition and gain footholds across global markets.

The silver lining here comes in the form of the hybrid cloud. Companies can continue to use their datacenters and still connect directly to the internet's backbone, while utilizing available interconnections that are available to bridge their systems with those of Amazon Web Services (AWS), Microsoft Azure, or Google Cloud Platform (GCP).

Before moving to the hybrid architecture

Under the previous approach, datacenters had to be built up to handle the next 6 to 12 months of anticipated growth, with massive upfront investments across servers, routers, and switches, not to mention redundancy planning and the electrical power to run and cool the infrastructure. If a team wanted to build up a new service, the lead time alone would be cause for concern. All of this would tend to restrict creativity and free thinking, because the lead time from day one of work to production would make most new ideas cost and time prohibitive.

Additionally, providing a holistic view of operations across all hardware, in the field and in the datacenter, ran the risk of data growth expanding beyond the capabilities of the hardware that was available. Even though the size of the platform-level sensor networks can vary from company to company, the torrent of data required to monitor service-level operations, network operations, system-level operations, and connectivity operations is on the order of many billion events worldwide every single day—and that is only the general, platform-level operational analytics.

Synchronizing state and maintaining consistent, predictable global data SLAs required infrastructure that was designed specifically for this kind of elastic scale and could flex to handle the ebbs and flows of common traffic patterns as well as the kinds of traffic that is more anomalous—like those of a natural disaster, or celebrity video going viral. As a compounding effect, slow ETL jobs and variable latency networks around the globe would restrict general data availability SLAs, and this caused many blind spots within the network.

Shifting to a hybrid architecture

Given that sensor networks lend themselves nicely to the event-stream paradigm, the first step is to define and agree upon the schemas that will define the data structures for the myriad sensor events. The next step simply being to emit the events into regionally distributed, cloud-hosted event buses like Amazon's hosted Kafka solution.

This process tends to happen over many months, and is managed much more effectively by initially piloting one end-to-end solution. This allows for learning about such important system requirements as overall record size, the number of records produced per day, as well as how to efficiently partition topics, and what primary key to use for even distribution. After a pilot, the findings can be wrapped up into best practice guides for other teams to move faster when following suit.

Benefits of the hybrid architecture. By migrating to a near-real-time, streaming event-based architecture (Figure 4-3) using a NewSQL operational database, the company was able to keep all of the proprietary software, chips, and hardware, along with all of the datacenter points of presence, while also taking a tactical approach to harnessing the power of the cloud, improving operations at a global level.

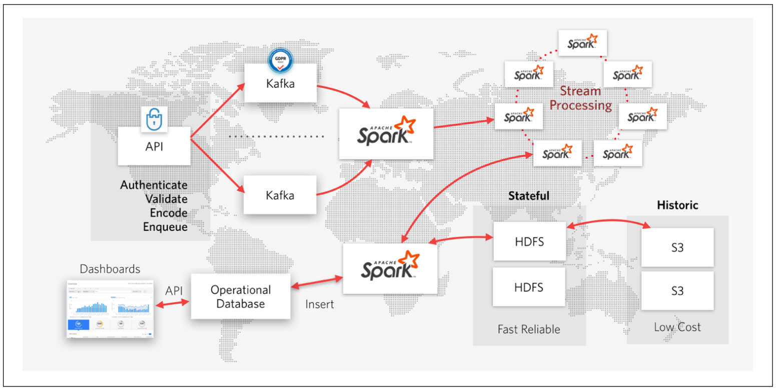

Figure 4-3. Hybrid analytics and insights architecture

This reduced friction by enabling executives, managers, support staff, and engineers to easily view and query data in a unified platform. Outages were resolved faster because data was generally available to help pinpoint which systems were under pressure or experiencing network unavailability (which typically occurs due to network partitioning). Root cause analysis for any number of problems was reduced from weeks of manual collection and analysis of data across countless disparate systems to sometimes less than an hour.

By operating off of the same data pipelines, teams could easily tap into streams of data to produce new specialized streams to power near-real-time monitors of critical business operations, to feed into real-time dashboards, and to provide further engagement and self-service capabilities for internal and external stakeholders alike.

CHAPTER 5

Selecting the Right Database for the Job

The Road Ahead

Given the explosive demands of todays' data-rich ecosystems, it is even more critical for businesses to look at the road ahead. This strategy of seeing the forest for the trees enables keen practitioners to observe the micro- and macro-trends that influence technology across industries, and it gives them a preemptive head start against the competition. It is common to have a roadmap that identifies the next one to three years in terms of broad architectural decisions, that is revised as needed but can be used as a blueprint for company-wide initiatives.

Given the many features and requirements to consider, as well as the sometimes nearly ideological level of conviction that people develop as to the correct approach, selecting the right tools for the data-driven company is a complex undertaking. Data and system architects must be aware of the myriad frameworks and technologies that can be used to provide critical data platform capabilities. These include distributed cluster computing frameworks, fault-tolerant event stream processing platforms, various elastic storage solutions, and, finally, knowledge and understanding of numerous database categories.

In the next section we break down the capabilities available across the common database categories and highlight the advantages of NewSQL.

Database Selection Criteria

Deciding which database technology to adopt is use-case driven, but as the capabilities matrix in Table 5-1 points out, a clear winner emerges with the NewSQL category, which enables a wide range of use cases, including the operational analytics use cases described in the sections that follow.

Table 5-1. Database capabilities

	OLTP	OLAP	NoSQL	NewSQL
Schema	✓	✓	×	✓
SQL	✓	✓	×	✓
Consistency	✓	×	×	✓
Availability	✓	✓	✓	✓
Replication	manual	manual	✓	✓
Security	✓	limited	limited	✓
Performance	✓	✓	✓	✓
Scalability	×	limited	✓	✓
Analytics[a]	×	✓	limited	✓

[a] Fast, robust support for a wide range of analytics use cases.

Selection Criteria for Operational Analytics

Selecting the optimal solution for your operational analytics needs is multifaceted, but ultimately can be looked at in terms of what you get with a given solution.

Integrations

1. Is the solution interoperable with your current tooling? Technologies? Operating system–level architecture?
2. Can it run on-premises? Across cloud providers? Are you locked into a single cloud vendor?
3. How flexible is data import and export? Does the solution provide pipeline connectors to common event streaming platforms like Apache Kafka?

4. What import-level guarantees exist? Exactly-once? At-least once? Is there a data availability Service-Level Agreement (SLA)?

Automation

1. How easy is it to operate the service?
2. Does it back up automatically?
3. Can the service be configured to be self-healing?

Extensibility

1. Does the solution support stored procedures?
2. Does the solution support user-defined functions (UDFs)?

Machine Learning Automation

1. Does the solution support machine learning automation such as automatic predictive analytics, forecasting, and retraining of machine learning models?
2. Does the solution integrate well with existing machine learning libraries and languages?

Security

1. Is the solution secure?
2. Does it provide end-to-end transport-level encryption?
3. Is the data encrypted at rest?
4. Does it support data access auditing?
5. Does the system connect to your company's Lightweight Directory Access Protocol (LDAP)?

NewSQL Offerings

As a means to simplify the process of selecting the correct NewSQL database for the job, let's now take a look at a couple of the available options within the NewSQL database market.

SingleStore is a cloud-native, highly-available, fault-tolerant, self-healing, ultra-consistent, fully distributed NewSQL database.

SingleStore provides fast performance due to being written from the ground up to perform online transaction processing (OLTP) and online analytical processing (OLAP) queries against both in-memory rows, and across efficiently optimized columnar storage that lives on disk. Their distributed query processor distributes work across the cluster in order to perform tasks directly against the data where it exists in memory, reducing bandwidth overheads and enabling linear scalability within the cluster.

In addition to the core database offering, SingleStore also provides ingestion monitoring, exactly-once ingest guarantees through their Pipelines API, security monitoring and auditing support, as well as full ANSI SQL support. This lowers the barrier to entry for analysts and engineers alike given that SQL is widely used across the indus-try. SingleStore can be run on-premises, in any cloud, or managed by SingleStore (SingleStore Managed Service).

Google Cloud Spanner

Cloud Spanner is Google's scalable, globally distributed, strongly consistent, transactional Database-as-a-Service (DBaaS). It automatically replicates data across global availability zones to provide near-zero-downtime (99.999% uptime guarantee), high availability, strongly consistent database with enterprise-grade security. Google Cloud Spanner is closed source and can be run only within Google's cloud.

CockroachDB

CockroachDB is inspired by Google Cloud Spanner. It offers a competitive distributed DBaaS that can be fully hosted and managed. CockroachDB provides high-availability from hardware and network fault tolerance, and geographically distributed data is stored in the same region where it was ingested. This simplifies some data governance concerns such as keeping data only in the region where a user produces that given data.

Additionally, CockroachDB offers enterprise-grade security through isolation of cloud-managed services within single tenant clusters by placing customers on their own dedicated servers. In addition to service-level isolation, it provides Transport Layer Security (TLS 1.2) for secure end-to-end connectivity and transport of data. CockroachDB is open source and can be run on any cloud provider or on-premises.

Decision Time

When it comes down to making the correct decision regarding what steps your company will take on the road toward operational analytical excellence, there are many factors to weigh. We have seen that there can be long lead times associated with building home-grown solutions, longer lead times to coordinate company-wide efforts, including moving toward new technology stacks, as well as ramp-up time and training for nontechnical staff to learn how to use new tools and capabilities. There are also huge gains to be made, such as the general unification of your overall business analytics platform, improvements in terms of overall platform—which include less time spent maintaining large monolithic software stacks and siloed databases, as well as increased agility in terms of how the company can approach new business opportunities, maintain current product offerings, and scale to meet expected and unexpected needs in the years to come.

About the Author

Scott Haines is a full stack engineer with a current focus on real-time, highly available, trustworthy analytics systems. He is currently working at Twilio as a principal software engineer on the Voice Insights team, where he helped drive Spark adoption and streaming pipeline architectures, and build out a massive stream-processing platform.

Prior to Twilio, he worked on writing the backend Java APIs for Yahoo! Games, as well as the real-time game ranking/ratings engine (built on Storm) to provide personalized recommendations and page views for 10 million customers. He finished his tenure at Yahoo! working for Flurry Analytics, where he wrote the alerts/notifications system for mobile.