O'REILLY®

Compliments of snowflake®

Designing a Modern Application Data Stack

Considerations for Scalability, Data Processing, and Application Distribution

Adam Morton, Brad Culberson & Kevin McGinley

REPORT

Snowflake

snowflake

BUILD, DEPLOY AND SCALE DATA-INTENSIVE APPLICATIONS WITHOUT OPERATIONAL BURDEN

snowflake.com/applications

developer.snowflake.com

Designing a Modern Application Data Stack

Considerations for Scalability, Data Processing, and Application Distribution

Adam Morton, Brad Culberson, and Kevin McGinley

O'REILLY®

Beijing • Boston • Farnham • Sebastopol • Tokyo

Designing a Modern Application Data Stack

by Adam Morton, Brad Culberson, and Kevin McGinley

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

- Acquisitions Editor: Aaron Black
- Development Editor: Melissa Potter
- Production Editor: Katherine Tozer
- Copyeditor: Paula L. Fleming
- Interior Designer: David Futato
- Cover Designer: Karen Montgomery

- Illustrator: Kate Dullea
- October 2023: First Edition

Revision History for the First Edition

- 2023-10-20: First Release

statement of editorial independence.

978-1-098-15749-4

[LSI]

Introduction

Data-driven products and services have exploded onto the market in a big way. According to a recent Foundry data and analytics study, 88% of IT decision makers say that data collection and analysis have the potential to fundamentally change the way their companies do business over the next three years. Companies leveraging huge volumes of data at scale have disrupted traditional business models, forcing former market behemoths to adapt or face irrelevancy. Data and analytics tools are expanding into and being embraced by new departments, such as product management and marketing.

The rise in data-driven initiatives is also powered by a significant change in customers' expectations. A decade ago, it was enough for a website to recommend products based on previous buying behavior. Now, customers look for a highly personalized experience that is tailored specifically for them—a level of customization that requires organizations to analyze vast quantities of data in order to satisfy their customers and gain a competitive advantage.

Internal customers' expectations are undergoing a noticeable shift as well. They now demand instant access to data, a clear understanding of provenance, and quicker delivery of valuable insights. Internal use cases are compelling catalysts for the development of more adaptable and widely distributed applications, and providing these tools quickly is essential to stay ahead of the curve.

The bottom line for both application vendors and customers is that data cannot be an afterthought—and data applications are becoming the standard for making that data accessible as well as distributing data-driven insights.

These data-intensive applications need a robust, high-performance data stack that can handle the variety and velocity of today’s data sources. The emergence of the cloud data platform ecosystem has allowed us to centralize a wide range of data formats within a scalable and flexible infrastructure, providing the ability to quickly and easily ingest data at low latency and transform the data by “pushing down” business logic. Centralizing the data also makes it easier to apply governance and privacy controls. We can visualize the data with interactive dashboards and bring analytical models to the data to generate insights faster than ever. The demand for flexible, on-demand data access has also sparked a shift in commercial model preferences, from paying for fixed capacity to usage-based pricing.

But what’s next? New capabilities developed to share data seamlessly both externally and internally are paving the way for the next iteration of the cloud data platform. We believe the next wave of innovation will see the modern cloud data platform move into the application space.

In response to this important trend, we created this report to inform and guide application teams as they design and build data-intensive applications at scale. We’ll show you how using a cloud data platform can help you design, build, deploy, and share data-driven applications between business units; better serve your customers; and generate new revenue streams by monetizing your applications.

In Chapter 1, you’ll learn about the capabilities that make cloud data platforms a great fit for data-intensive applications. In Chapter 2, we take a look at important considerations for building applications at scale, such as resource sharing, multi-tenancy, workload isolation, and the separation of compute and storage. In Chapter 3, we discuss how to process and distribute data with a focus on efficiency to keep your application as simple and manageable as possible.

With a better understanding of data-intensive application architectures on cloud-based data platforms, you can design a modern application data stack that takes full advantage of advances in data processing and app distribution to accelerate development, deployment, and adoption cycles—allowing you to provide business value to your users and customers faster.

Chapter 1. The Cloud Data Platform Is a Great Fit for Data-Intensive Apps

Today’s data applications are remarkably different from their predecessors. The world of applications has witnessed a paradigm shift, where the ability to handle and process huge volumes of data rapidly has become a major focus. It is within this shift that the cloud data platform has emerged as a compelling solution tailored to address the unique challenges posed by data-intensive applications.

This chapter explores the symbiotic relationship between the evolving demands of applications and the features of cloud data platforms, demonstrating why these platforms are perfect for building data-intensive applications.

Benefits of a Modern Cloud Data Platform

As companies migrate from on-premise infrastructure to the cloud, they gain several benefits, including:

- Access to virtually unlimited computing resources
- Improved reliability, availability, scalability, and security

- Flexibility to strike an optimal balance between performance and costs

However, the evolving needs of data applications have gone beyond these benefits, spurred by the modern data platform. Modern applications demand a comprehensive set of functionalities to effectively operate, including the ability to collect, analyze, and manipulate large volumes of data in microbatches or in near real time. Additionally, these applications require secure handling of diverse data types and structures, seamless integration with external tools and data sources, a data-sharing network to facilitate application distribution, and efficient scalability to meet customer demands without over- or under-provisioning resources.

The modern cloud data platform combines these factors, providing a single source for processing, analyzing, and extracting value from vast amounts of data. It removes traditional roadblocks to sharing and accessing data, facilitating innovation and data collaboration. Let’s take a closer look at a few reasons why a modern cloud data platform makes an excellent foundation for data-intensive apps:

Elastic scalability
This allows you to automatically adjust resources to meet fluctuating demand. Elastic scalability aims to ensure your cloud platform can grow with you, handling future data needs without compromising performance or incurring excessive costs. Dynamic allocation capabilities allow the system to add or remove resources as demand changes, while load

balancing evenly distributes work across available resources. This flexibility is complemented by pay-as-you-go models, where the system minimizes costs by scaling down during periods of low demand.

Reduced operational overhead

Platforms offer managed services that handle hardware and software maintenance, updates, security, and customers’ costs associated with managing infrastructure.

Availability

In the cloud, data is automatically replicated within a geographical location called an Availability Zone (AZ). This approach safeguards the data from individual data center failure. Your disaster recovery plan may also require you to replicate data between AZs, an approach known as *Multi-AZ*, or across cloud providers to minimize data loss and ensure maximum application uptime.

Monitoring

As part of your app data stack, a cloud data platform can monitor logs and metrics and send notifications when thresholds are reached or significant events occur. Monitoring enables your solution to recognize and automatically recover when it crosses low-performance thresholds or experiences failures.

Alerting

One of the primary reasons you monitor is to effectively detect failures. But if you monitor *without* alerting, you may as well not monitor at all. Decouple your alerting from your systems as much as possible; after all, if a service interruption removes your ability to alert, you will have a longer period of interruption. You never want your customers to know about problems before you do.

Near real-time processing
Modern cloud data platforms should include processing services that can handle continuous data streams, perform real-time analytics, and trigger immediate responses. These capabilities are crucial for use cases such as real-time analytics, fraud detection, recommendation systems, and Internet of Things (IoT) applications.

Cloud Platform–Specific Considerations

We’ve covered a lot of benefits to help you understand why a cloud data platform is a great fit for applications, but there are several key considerations to keep in mind when selecting a platform to support your workloads.

Cross-Cloud Deployment and Application Development

Many organizations have numerous applications and data sources scattered across multiple cloud platforms. This scenario often leads to teams investing significant time in consolidating, preparing, and managing data. As Figure 1-1 illustrates, the cloud data platform is included on top of the underlying cloud providers’ services, providing a level of isolation. This means clients can integrate workloads from systems residing on different cloud platforms with minimal effort.

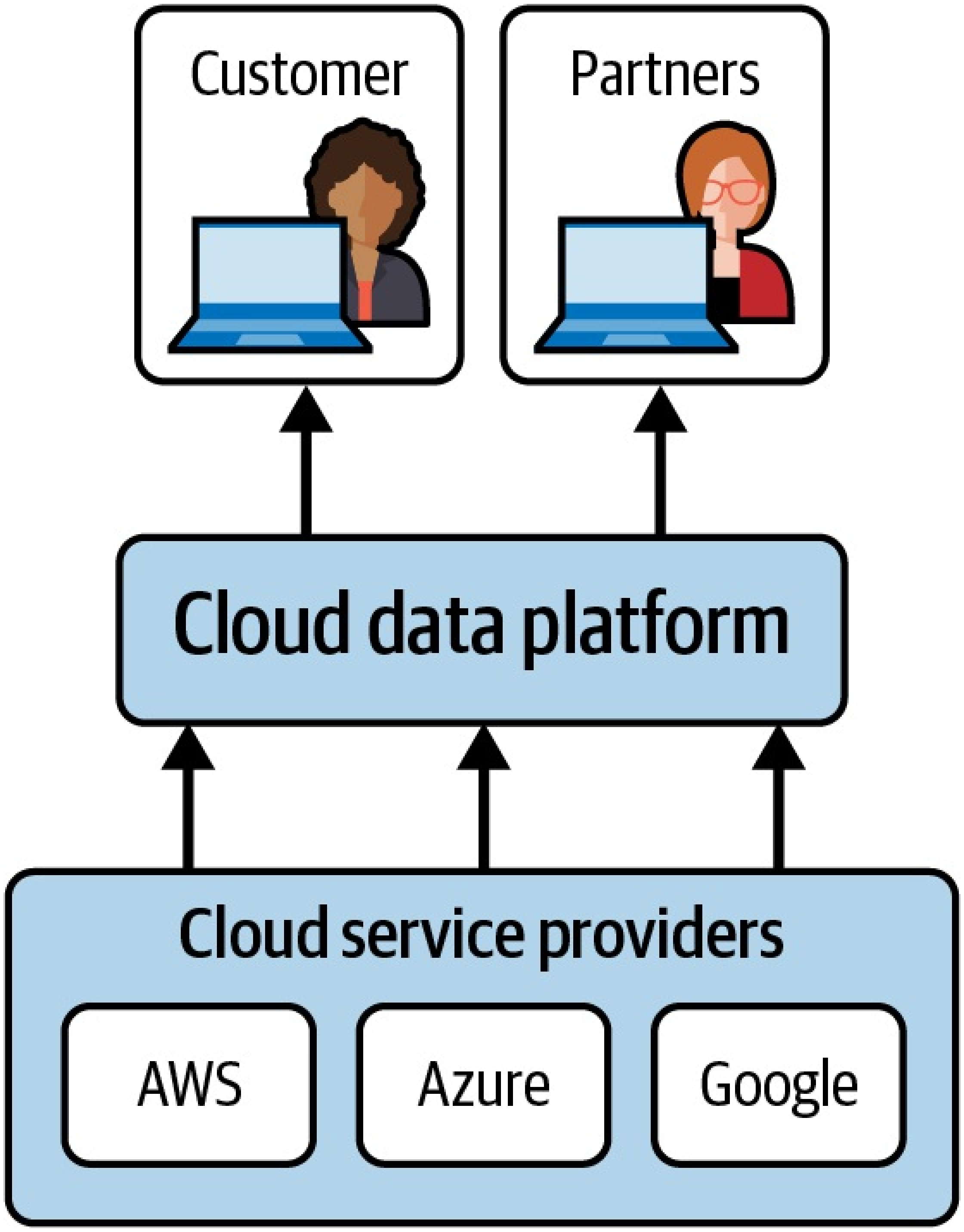

Figure 1-1. Leveraging a cloud data platform consolidates workloads across cloud providers.

A cloud data platform must deliver abstraction at the data layer so the user experience is consistent regardless of which cloud provider is hosting the data or application. This shields you from needing to get into the complexities or nuances specific to each cloud provider.

Adopting a cross-cloud deployment strategy gives organizations access to several benefits when building applications:

Cost-effectiveness
: You can leverage the pricing models, discounts, and offerings of multiple cloud providers. This enables you to select the most cost-effective options for different components of the application or service. You can take advantage of competitive pricing, specific features, or regional availability offered by different providers, ultimately optimizing costs.

Resilience and availability
: If one cloud provider experiences an outage or service disruption, you want to be able to failover to another cloud platform and redirect your application workloads without your users ever knowing. This strategy minimizes downtime and ensures continuity of service for your business. Introducing this redundancy across cloud environments safeguards against single points of failure and enhances the overall reliability of your application or service.

Effective workload distribution
: Access to multiple clouds allows you to deploy your app in proximity to end users. You can also factor in data locality, consumer cloud preference,

and/or specialized services offered by specific providers. This approach can result in improved performance, reduced latency, and scalability options tailored to specific regions or groups of customers.

The advances around cross-cloud deployment present new options for how you approach your application data stack. Your teams can build applications that access and use data from any public cloud in any geographic region. You also have options related to the ownership of data, which we'll explore next.

Application Deployment Models

With the increased availability of modern data platforms, companies of all sizes can adopt massively scalable and cost-efficient data platforms that are on par with those of their vendors. This gives application builders and providers new options for how to deploy their applications: as managed or connected applications.

Software as a service (SaaS) applications typically store and process data in a data store managed by the SaaS application provider. This more traditional application deployment model is known as a *managed app*.

With a managed app deployment model, customers provide the SaaS application providers access to ingest the customers' data so it can be processed within the providers' own, dedicated data platform. This means each application effectively creates a new silo (see Figure 1-2), which can lead to a fragmented data landscape and hold back customers' efforts to become data driven.

The emerging *connected app* model, however, allows application providers to connect into the customer's account, keeping the customer's data within the customer's control. Also, the customer can use the application, and the provider doesn't need to worry about exposing their code to the customer, as shown in Figure 1-3.

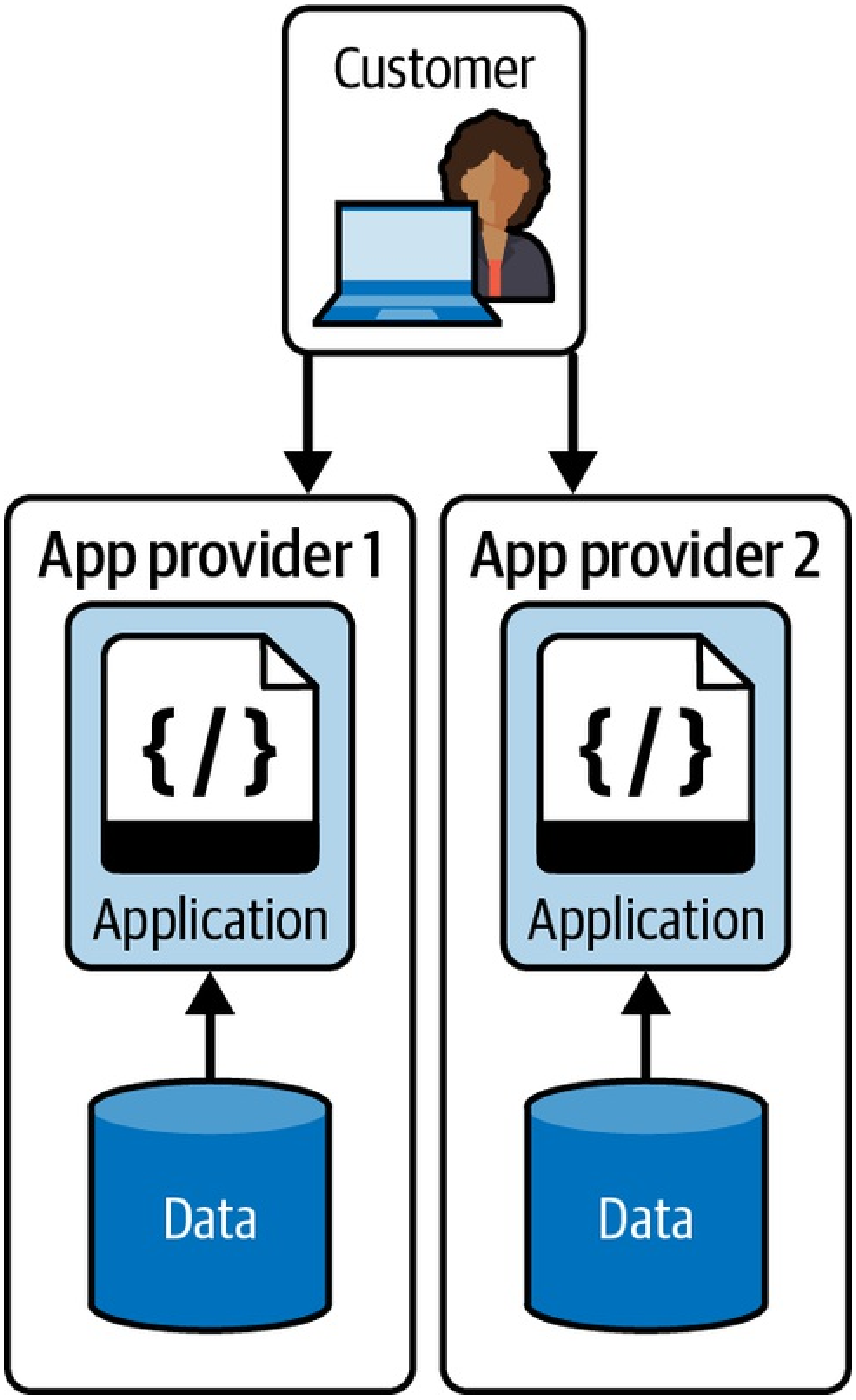

Figure 1-2. In a managed app model, customers hand over data ownership to the app provider.

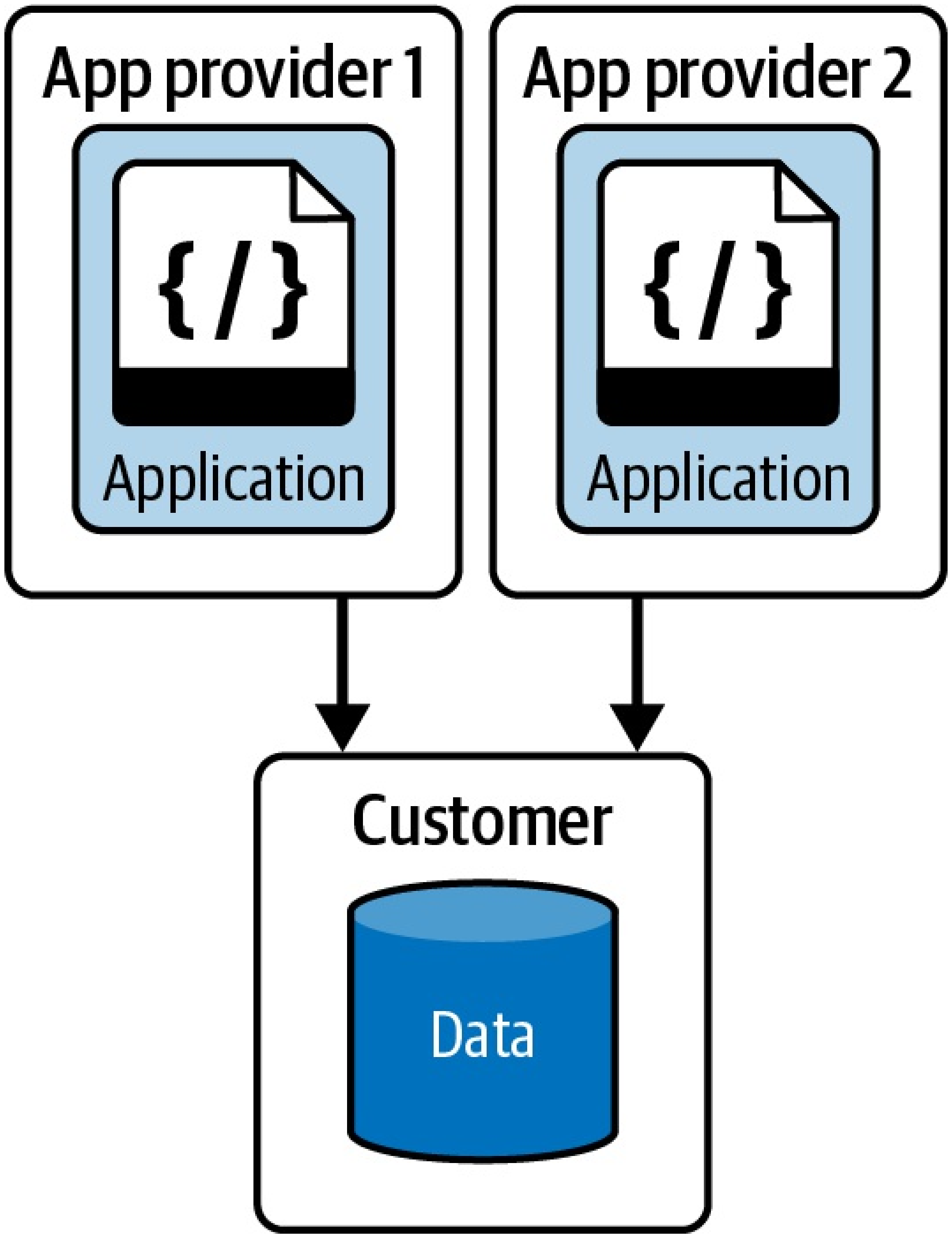

Figure 1-3. Customers maintain ownership of data in a connected app deployed within the cloud data platform.

Customers may opt for the connected app model if they have regulatory obligations (such as the handling of personally identifiable information or data sovereignty laws) that require them to keep their data within their own environment. All of the data can be stored and processed on the data platform while maintaining existing security controls and governance. The connected app model also simplifies the commercial discussion because the customer is using their own resources, resulting in an easier economic conversation between both parties.

This approach can also help reduce data silos created by individual SaaS applications and make it easier to govern and manage sensitive data, as the application code is independent of the customer’s data.

NOTE

In the connected app model, communication and coordination between customer and provider are key. The customer should check with the provider before making changes to configuration settings to avoid any impact on the app’s performance or security.

Remember that choosing your application deployment model is not an either/or situation. Application providers can offer both a fully managed experience and a connected application model, giving customers the flexibility to choose the model that best fits their specific requirements.

Extensibility

Extensibility is crucial in a cloud data platform, enabling application builders to work efficiently by supporting various programming languages. While SQL is typically at the core of modern data platforms, relying solely on SQL becomes limiting as platform capabilities expand.

To empower data consumers, it is vital that the platform allow them to extend its native functionality through preexisting libraries and tools, streamlining their data workflows. By providing comprehensive language support for Java, Python, Scala, and more, the platform accommodates a diverse user base, allowing customers to work in their preferred language for data manipulation, analysis, and application development.

Some of the tools a cloud data platform provides to support this extensibility are:

User-defined functions (UDFs)
You can write a UDF in SQL, Python, JavaScript, or Java. Data engineers would select the best language to author a UDF and encapsulate specific business logic within it before creating it on the cloud data platform. This allows other users and applications to call the UDF while promoting reusability of code and consistency of business logic.

Stored procedures
Stored procedures encapsulate coded business rules or logic. They can also cater to error handling, dynamically build up a SQL statement before executing it, and allow for branching and looping—none of which are

supported by standard SQL.

External functions

You can use external functions to access external API services, such as machine learning models or geocoding functionality. You can pass data from your cloud platform to an external function, obtain a result set, and write this back to the cloud data platform.

Interactive web applications

You can create interactive dashboards and machine learning web apps by using programming languages more familiar to your data engineers, such as Python. This reduces their reliance on other teams for frontend app dev skills, improving productivity and allowing teams to get data in front of business users faster and reduce time to market.

Security

When a customer is entrusting you to store and process their data, extensive security is nonnegotiable. Data-intensive applications may also deal with sensitive or regulated data, heightening the demand for security even further. A modern cloud data platform should provide robust security measures, including encryption, access controls, audit logs, and compliance certifications standard and out of the box, so you don’t need to configure them each time you deploy your application. These features contribute to the application’s security and governance:

Authentication

The platform should offer strong authentication capabilities to verify the identities of users accessing the data. This includes support for various authentication methods such as username/password, multi-factor authentication (MFA), key-pair authentication, or a federated logon using OAuth or single sign-on (SSO) leveraging an identity provider.

Access control

Role-based access control (RBAC) is crucial for managing user permissions and defining access levels within the platform. It enables administrators to assign specific roles to users or groups, granting appropriate access privileges based on their responsibilities and job functions.

Fine-grained security controls

Frequently you'll come across use cases where you must give certain users access to sensitive data while protecting it from being viewed in plain text by unauthorized users. It's important that the platform allow for fine-grained security where you can ingest and store data once and apply a layer of controls, which, at execution time, uses logic to determine precisely what data to return to the user or application running the query.

Data encryption

A modern cloud data platform should offer robust data encryption capabilities. This includes encryption of data at rest and in transit.

Encryption safeguards the confidentiality and integrity of data, making it unreadable to unauthorized individuals even if the data is compromised.

Summary

In this chapter, you learned that a modern data platform is a great foundation for data-intensive apps due to its elastic scalability, which provides you with the ability to scale up and down to match the demand profiles of your customers. This ability is more important than ever as data volumes continue to grow exponentially.

You’ve learned how being able to operate seamlessly across clouds is changing the nature of application development, easing the deployment pathway and providing a way to deliver a consistent user experience regardless of which cloud provider is powering your cloud platform. The connected application model, made possible in part by the rise of cloud data platforms, provides flexibility in deployment and data ownership that can benefit both app providers and customers.

The ability to extend the out-of-the-box capabilities of your cloud data platform allows you to expand the types of use cases your platform can support. Your teams can take advantage of all your data without extracting it out of the cloud platform and into separate tools.

In the next chapter, we focus on the tools a cloud data platform provides to help you overcome common challenges of running your application at scale across multiple customers.

Chapter 2. Scalability

Striking a balance between affordability and quality is crucial, especially for cost-conscious customers. This chapter explores the importance of scalability when building apps on top of a cloud data platform, focusing on the challenges of accommodating diverse customer needs and the costs associated with automation for application developers.

Different industries have distinct application requirements that significantly influence the architectural approach you should adopt. The functionality of your application will be guided by its target addressable market, encompassing both external- and internal-facing applications. For example, a financial application catering to hedge funds may not have the same customer base as a website builder serving millions of users.

Market size plays a crucial role: some industries have smaller markets with customers willing to pay a premium for specialized services, while others have larger markets with expectations for lower prices. Additionally, it’s important to tailor internally facing applications, such as those used for internal operations, employee management, or data analysis, to meet the unique demands and workflows of the teams involved. Understanding these dynamics is essential for crafting applications that not only cater to external customer needs but also streamline internal processes, ultimately enhancing efficiency and effectiveness within your organization.

Single Versus Multi-Tenancy

Tenancy refers to the way the system allocates and isolates resources to accommodate the needs of different applications. In this section, we’ll focus on how you can make resources available to support the users of your application, as well as how their data may be segregated to prevent unauthorized access or data leakage.

There are two primary options when it comes to tenancy:

Single tenant
Each customer (tenant) has a dedicated instance of resources.

Multi-tenant
Multiple tenants coexist and utilize cloud resources without access to each other’s data.

The single-tenant model offers certain advantages because the provider exclusively allocates compute resources to the customer and stores one customer’s data separately from that of all other customers (tenants). Upgrading or customizing the environment poses no risk to other customers, providing enhanced security and flexibility. Table 2-1 summarizes the differences between the two models.

Table 2-1. Single-tenant versus multi-tenant configurations

	Single-tenant	**Multi-tenant**
Isolation	Resources completely isolated for each tenant, providing better security and compliance	Shared resources, requiring strict security policies
Customization	Highly customizable for individual data needs	Limited customization, as data model changes affect all tenants
Performance	Predictable—no contention with other tenants	Potential contention—requires constant monitoring
Cost	Typically more expensive	More cost-effective due to shared resources and bulk processing
Scalability	May require more planning	Generally easier to scale

Complexity	Easier to manage individual tenants	More complex due to shared resources
Security	Easier to ensure through isolation	Requires careful management for secure data isolation

While single tenancy has desirable benefits for all customers, maintaining a single tenant for each customer can become prohibitively expensive for application vendors operating at a large scale. Automation and streamlined processes are required to effectively manage and serve numerous tenants, considering the cost constraint associated with fixed overhead per tenant. This cost dilemma often leads organizations to adopt a multi-tenant configuration.

Figure 2-1 illustrates the fundamental differences between single- and multi-tenant architectures.

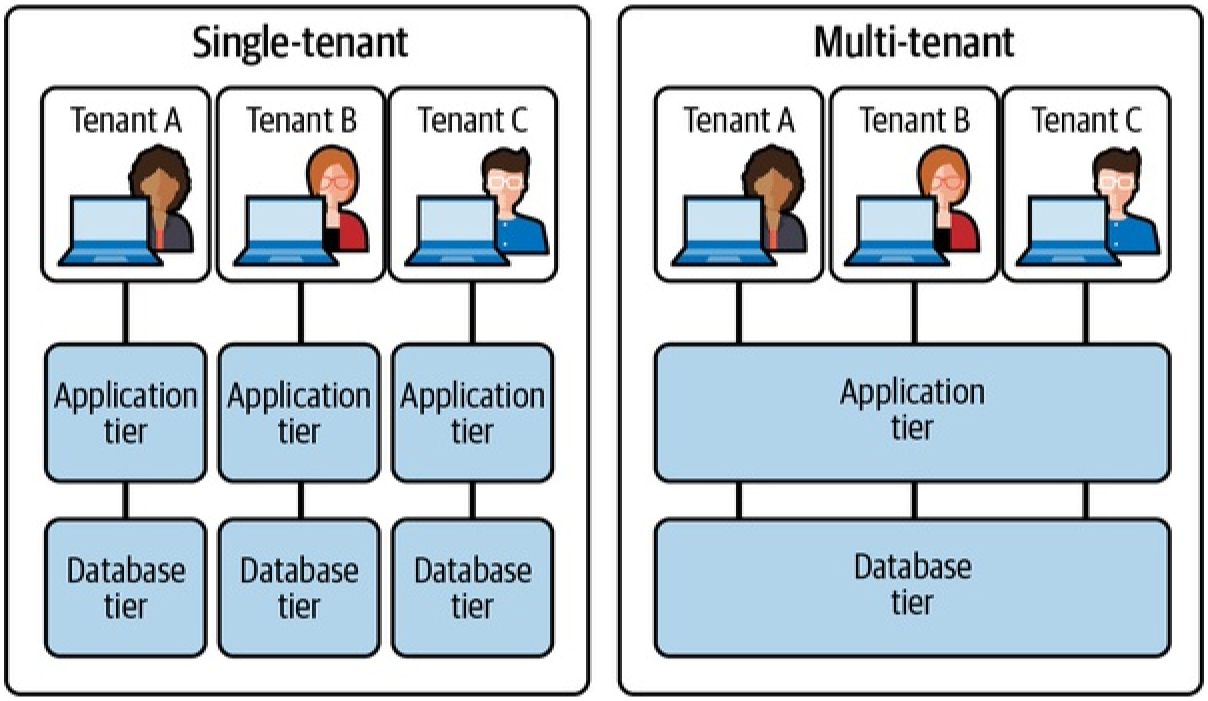

Figure 2-1. Single-tenant versus multi-tenant environments

NOTE

A single-tenant model provides isolation for customers but can increase costs and complexity for the provider. A multi-tenant architecture can reduce the provider's costs and complexity, but requires careful design to maintain isolation and performance.

In a multi-tenant model, resources can be shared and pooled among tenants, allowing application vendors with a large customer base to optimize efficiency and reduce costs. However, the challenge of the "noisy neighbor" problem arises, where the performance of one tenant is affected by the activities of another. Mitigating this issue involves careful workload isolation to ensure fair resource allocation and promote a balanced multi-tenant environment.

Nevertheless, some customers value the security and control provided by single tenancy enough to pay a premium for their dedicated environment. This is much like paying a toll to use a faster road to reach your destination. Some drivers will be OK using the road with heavier traffic; others will wish to pay more to get there faster.

Market segmentation plays a crucial role in determining the appropriate tenancy model, and product offerings must align with the specific demands of customers operating within their respective markets. In the next section, we will explore effective resource and workload management in a multi-tenant environment.

Resource Sharing and Workload Isolation

For application developers, delivering an optimal customer experience while balancing performance and cost is crucial. We’ve learned that tenancy plays a significant role in managing cost and performance. A delicate balance must be struck between isolating workloads for security and optimizing compute and resource efficiency.

Regardless of which tenancy model you choose, a data-intensive application will run multiple types of workloads. Workload isolation means you can reserve resources exclusively for a specific type of workload, with the aim of eliminating resource contention between workloads.

To deliver clean, well-governed data in an optimized format, an application will typically have data ingestion, processing, and serving workloads. These workloads put demands on resources and require efficient management. As we will explore in detail later in this chapter, you can configure resource pools with specific characteristics that separate and isolate different processes to align with each workload's requirements. This not only minimizes the risk of contention and improves performance but also allows you to track the costs associated with each workload. Consequently, you can easily provide transparent pricing specific to each customer and workload, facilitating clear and accurate cost allocation.

Best Practices for Building a Scalable Data Application

In the past, on-premise systems tightly coupled storage and compute to minimize latency between data reading and writing. However, advancements in networking and storage bandwidth, coupled with decreasing costs, have changed this landscape. Increasing storage demands have created a need to scale compute resources, even if user or application requirements don't warrant it. This has resulted in unnecessary costs and inefficient resource utilization.

To address this, it is essential that your cloud data platform decouple compute and storage resources. Decoupling allows independent scaling of each layer based on specific workload needs. By scaling compute and storage resources independently, you can avoid waste, maintain optimal workload performance, allocate resources efficiently, and minimize unnecessary expenses.

Let’s explore some best practices for scaling compute and storage resources to optimize performance and cost-effectiveness.

Storage Considerations

Compared to compute, storage is relatively cheap, especially when taking into account compression. You won’t be under pressure to free up storage unnecessarily by purging data that may become valuable in the future.

It’s also important to consider how caching can help improve performance. Not caching access to resources, such as connections to your data platform or data, impacts performance, directly affecting your users. Consequences include:

- Making excessive calls to open database connections from the application to the data platform
- Fetching the same data repeatedly from the storage layer

If you have data that can be shared, either among tenants or among users within a single tenant, it’s likely that caching the data will help to reduce the load on your data tier. This will reduce costs and latency while improving performance and the customer experience.

NOTE

Spend some time thinking about the data in your own environment to develop your caching strategy. Good candidates for caching include data that changes relatively infrequently but is regularly requested by your application.

Next, let's delve into a few approaches to scaling resources.

Vertical Versus Horizontal Scaling

In traditional on-premise systems, administrators often had to plan for downtime and physically move or replicate data to other servers to accommodate increased user demand and queries. However, with cloud data platforms, your data is securely stored, organized, and accessible on the platform, so you can focus on optimizing your compute resources to efficiently leverage your data.

An important aspect of your cloud data platform is the availability of stateless compute resources. These resources do not store any data between requests, allowing for rapid and automatic scaling without the concern of losing persistent data. From a scalability perspective, stateless components enable quick addition or removal of compute resources on-demand. This flexibility empowers you and your teams to respond almost instantly to the changing needs of your customers or business, all without downtime.

The isolation of storage and compute means that you can scale an existing set of compute resources vertically or horizontally. *Vertical scaling* involves increasing the power and size of your existing server(s). This is sometimes called “scaling up.” *Horizontal scaling* means increasing the number of servers alongside your existing server(s). This is sometimes called “scaling out.”

Along with vertical and horizontal scaling, you can also spin up a brand-new pool of dedicated resources and redirect your queries to use it. As an application builder, you should factor into your selection criteria how quickly a cloud data platform can provision more, or new, compute resources.

When deciding on an approach to scaling, it’s important to consider which workloads will respond best to which scaling mode. With compute resources, for example, vertical scaling helps to support larger, more complex queries. Horizontal scaling is aimed at improving concurrency. This will really help when you have a large number of concurrent users and/or queries.

What we’re really looking for here is improving throughput, or the rate or speed at which our application can handle a certain amount of data within a given period of time. Throughput can be affected by both concurrency and latency. In certain situations, postponing the requirement for horizontal scaling becomes a viable option when vertical scaling can expedite query processing, thereby preventing queuing caused by slower queries. It is crucial to consistently contemplate both scaling models in tandem.

With your compute resources, your cloud data platform should make it easy to configure autoscaling mechanisms that adjust the demand on your system within your predefined parameters.

Be sure to consider tenancy when planning for scale. In a multi-tenant environment, as the number of tenants increases, you might need to scale your compute vertically to process data faster and avoid queuing. Additionally, to support a growing number of tenants sharing computing resources for data serving, you may have to scale horizontally to support the sheer number of queries submitted by tenants.

Workload Isolation

Workload isolation is a critical component for cloud data platforms supporting data applications, especially because the challenges of resource contention expand in multi-tenant environments.

In on-premise data warehouses, multiple workloads often compete for the same resources. For example, extract, transform, and load (ETL) processes are generally resource intensive, requiring large amounts of CPU, memory to process and store data, and I/O and network bandwidth when writing and reading data to the storage layer.

If data ingestion and user queries share the same hardware, a large data ingestion event could saturate the compute resources and cause poor performance for user queries. The workaround has been to run the ETL activities outside the business day so they didn’t impact the users and applications running queries. Today we have data arriving far more frequently, making this traditional and rigid approach impractical.

Your cloud data platform should enable you to establish clusters of designated resources that can be aligned with tasks and workloads of a similar profile. It should also enforce transactional consistency so that data is never in an invalid state when multiple workloads are running on the same data concurrently.

In general, the apps that support business-critical activities get more compute resources set aside for them. Although this assures enhanced performance, you must be mindful of the trade-off, which is a higher cost.

Row-Level Security

If the shapes of the data for each customer in a multi-tenant environment closely resemble one another, you would not want to create and maintain duplicate copies of the same data structures for each tenant. Not only does using a single data structure avoid unnecessarily duplicating data, but it also makes it easier to manage and control one set of data stored centrally. This is where row-level security can help provide robust controls.

Row-level security operates as an access control mechanism that offers fine-grained control over rows of data within tables. It enables you to create policies that apply conditions at execution time based on the role requesting the data (see Figure 2-2).

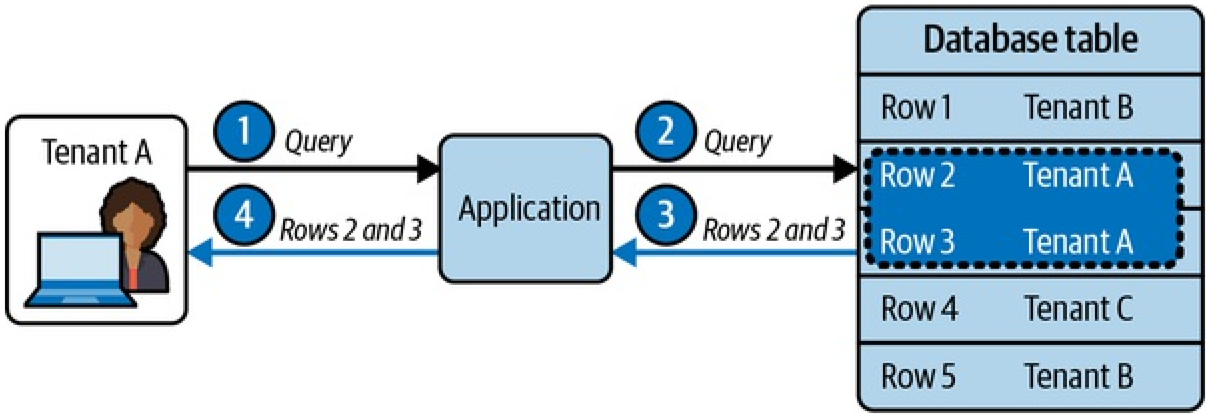

Figure 2-2. Process for applying row-access security in a multi-tenant environment

Figure 2-2 illustrates how these policies determine whether a given row in a table or view can be viewed:

1. Tenant A executes a query (via the application).
2. The application sends the query to the database, applying the user’s access restrictions.
3. The database returns only the rows that correspond to tenant A (rows 2 and 3 in this example).
4. The application sends these rows back to tenant A.

When designing an application for multiple tenants in a multi-tenant environment, it’s crucial to ensure that each tenant can access only their own data. By implementing row-level security, developers can guarantee that a tenant won’t be able to view or modify data belonging to another tenant. You can configure this security measure to automatically filter rows based on the tenant ID of the logged-in user, reinforcing data isolation and minimizing the risk of data breaches.

Row-level security also helps streamline the task of designing and maintaining your applications data, particularly in a multi-tenant environment. Data for each tenant can be stored within the same tables and have secure, reliable, robust controls to prevent cross-tenant data leakage. With row level policies, you can establish rules that ensure the segregation of each tenant’s data rows from those of other tenants.

Summary

This chapter emphasizes the significance of scalability for apps built on cloud data platforms. As different industries present diverse requirements, it’s essential to understand the specific needs of a target market, ensuring high customer satisfaction and platform viability.

Make informed decisions about the tenancy model, keeping in mind both scalability and security. Regularly reassess and optimize your resource management strategies, especially when working in multi-tenant environments. Adopt best practices like workload isolation and row-level security to ensure smooth operations and the utmost data protection. Taking all of these considerations into account when designing your applications will help you strike a balance among scalability, cost, and security.

Chapter 3. Efficient Data Processing and Distribution

Successful data applications deliver tangible insights to customers in the most efficient way possible. Modern data applications need to tap into an array of rapidly changing data sets and data formats while supporting a distribution model that delivers a consistent user experience.

It’s important to consider how to ingest and integrate data by building simple data pipelines that are easy to maintain and extend over time. In this chapter, we will look at how a cloud data platform allows you to reduce data movement and improve timeliness to deliver data to your application at scale.

Key Considerations for Data Processing

Processing data at scale presents a significant and complex challenge for many data teams. The primary objective of the data processing layer is to construct pipelines that efficiently and rapidly transfer data from source systems to the cloud data platform. These pipelines should be automated and resilient. They also often trigger subsequent processes that apply transformations to cleanse and standardize data, ensuring the consistent delivery of high-quality data in the required format. This gives your teams fast access to ready-to-use data, allowing them to focus their efforts on application development rather than maintaining data pipelines.

Let’s take a look at key considerations for efficient data processing, including data format, how to reduce data movement to improve data integrity and timeliness, and the ability to act upon changes in data as quickly as possible.

Data Format

We store raw data in its original format without applying additional business logic. A reliable cloud data platform should not only support structured data but also ingest semistructured data like JSON without additional preprocessing. This schema-on-read approach reduces latency and keeps the data pipeline intact, even if the JSON payload changes in the future. It’s important to note that query performance may not be as effective for raw data. Regardless of the format, raw data doesn’t perform as well as conformed data because you have to apply business logic on the fly with every query.

The difference between raw and conformed data has significant implications for standardization and efficiency. *Conformed data* adheres to a predefined format, promotes interoperability, and facilitates the consistent integration of data from multiple sources using standard business rules. This standardization enables the data platform to optimize data storage and allows the query optimizer to retrieve data more efficiently, resulting in improved performance compared to raw data. However, conforming data requires understanding the raw data and applying business logic, which involves time and effort as well as designing, building, and testing data pipelines. Additionally, any changes to the raw data feeds increase the risk of breaking the data pipeline and necessitate rework. These additional steps cause delays in delivering the data, whereas raw data allows for almost immediate action.

The choice between raw and conformed data depends on a thorough assessment of the business use case. For example, in situations where immediate notifications are crucial, such as when a weather event damages a railway bridge, using a narrow set of raw data while applying light business logic on the fly (instead of precomputed business logic in batch) is essential to promptly notify the teams on the ground. Conversely, if the goal is to plot physical infrastructure on a map with contextual information, the speed of data delivery becomes less critical, while the ease of integration and augmentation with other sources becomes more valuable.

Reducing Data Movement

It is more efficient to keep data in one location than to move it between systems. Moving data incurs costs, including storage and compute, latency, increased points of failure, complexity, and the need to build and maintain additional pipelines. Streamlining data movement within your overall solution is crucial for optimal performance.

Your cloud data platform should provide you with the necessary tools and connectivity to perform as much processing as possible within the platform itself.

In Chapter 2, we discussed ETL processes, which involve extracting data from its source, transforming it, and loading it into a conformed schema on the data warehouse for accessibility by applications and users. This process has traditionally required dedicated on-premise hardware to ensure resource availability and reduce workload contention.

However, leveraging the resources of the cloud platform now allows us to extract and load data in its raw format onto the platform before applying transformations in place to conform the data. This extract, load, and transform (ELT) approach offers numerous advantages and opportunities for efficient data processing, including:

- Removal of additional dedicated hardware and/or software to support ETL processes
- Faster access to raw data in its original format without impacting the performance of the system that generated it
- No data loss due to failure and better auditability between raw and conformed data structures
- Ability to invest time and effort to conform only the subset of data that provides real business value, along with the flexibility to go back and evolve that subset as business needs change

The transition from transforming data during its journey to the data warehouse to performing transformations on the cloud platform represents just the initial step in pushing down logic to where the data resides. Now we have the capability to extend this approach to include data visualization tools, artificial intelligence and machine learning models, and application logic. By pushing this complex processing logic to our cloud data platform, we can leverage its elastic storage and computing capabilities, in addition to the benefits we discussed in Chapter 1. It's important to highlight that, in the future, your cloud data platform should be actively progressing toward moving more processing tasks closer to the data. This includes transactional processing and various other essential services.

Data Integrity and Timeliness

Here we explore best practices for designing data pipelines to ensure timely delivery of data to your applications, users, or customers while avoiding unnecessary delays. Factors such as cost, performance, and data latency are crucial to consider based on your specific use cases.

While the availability of compute resources on your cloud data platform provides flexibility, it is important to be mindful of the associated costs. Generally, more powerful resources for data processing result in higher expenses. Instead of aiming for the same performance level for all customers, consider a tiered model (such as bronze, silver, gold) with different service level agreements (SLAs) for performance and latency at corresponding price points.

Another approach to reduce data latency is to process data more frequently. Running data pipelines as data arrives helps keep the data on the platform up-to-date. However, this requires the ability to isolate transformation workloads from serving workloads with ACID (atomicity, consistency, isolation, durability) compliance to avoid query collision and guarantee query results. If providing data more frequently does not bring significant value to the application users, it will be harder to justify the expense compared to refreshing the same data on a less frequent basis.

A core principle in efficient data processing is to process only the data that has undergone changes, an approach often referred to as *incremental processing*. When aiming to reduce costs and latency, it is crucial to have pipelines capable of processing only the data that is truly necessary. For example, if you have a source table with 10 million records and only 10% of that table experiences changes each day, you would want to process 1 million records rather than all 10 million.

The process of identifying changes in data is commonly referred to as change data capture (CDC). Due to the diverse range of source systems available, CDC technologies come in various forms to carry out the process of identifying these deltas at the source before they reach your platform. Figure 3-1 illustrates the process used to capture changes made at the data source and apply them to a target table.

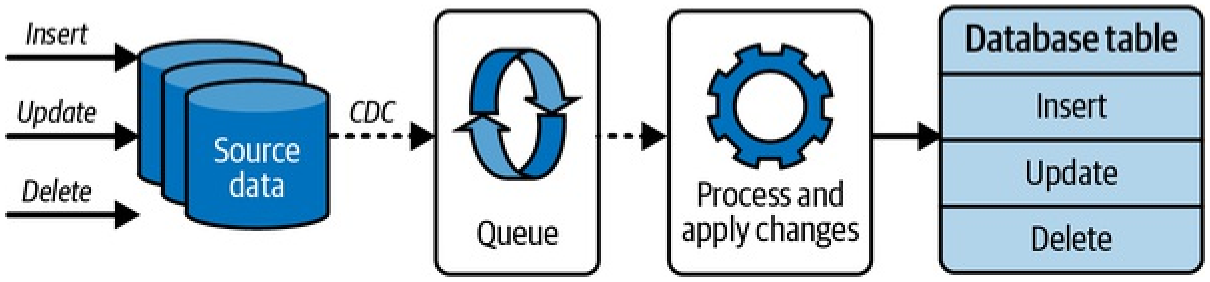

Figure 3-1. A typical CDC data flow

The CDC process is very similar regardless of the technology used:

1. A process monitors the source data to track changes as they occur.
2. When a change (insert, update, or delete) is made against the source data, it triggers the next step in the CDC process.
3. The next step in the process applies the change and adds metadata—typically the whole row plus an indicator to identify the change type (insert, update, delete) along with a timestamp. The changed record and its metadata are added to a table or queue ready to be processed.
4. Your data pipeline picks up the most recent changes from the queue/table and processes them on your data platform before flushing the queue for the next run.

Having a ledger of changes available for you to process gives you complete control over how and when you want to process changes and bring them into your platform.

Considerations for Distribution

The value of sharing and distributing data is enormous. A Gartner report estimates that data and analytics leaders who share data externally generate three times more measurable economic benefit than those who do not.

As you saw in Chapter 2, it’s important to be able to share your application with customers at scale while minimizing operational overhead. In this section, we’ll get into more detail about options for distributing your app, taking into account where to process data and how to securely share data with your partners or customers.

First, we will explore secure methods of sharing data with third parties within your ecosystem, including partners and customers. Then, we will examine the distinctions between hosting the processing yourself and delegating it to your customers’ data and how this can factor into your app design decisions.

Secure Data Sharing with Third Parties

Traditional approaches to data sharing pose significant obstacles for organizations. When you rely on outmoded data sharing practices like emailing spreadsheets or utilizing batch processes such as FTP, you may create outdated copies of data that are difficult to secure and audit, costing additional time and money. Data-driven organizations need a method that allows them to share data more efficiently and securely, eliminating data copying or movement and transforming data for usability.

A modern cloud data platform should enable you to securely and seamlessly share data with your customers and partners without them directly accessing your environment. This not only eliminates typical data-sharing challenges related to data silos and physical data movement, but it also provides a host of other benefits, including:

Real-time collaboration and access

Modern cloud platforms enable multiple users to access and work on the same dataset simultaneously. This ensures that everyone is working with the most up-to-date information.

Enhanced security and compliance

Cloud platforms often come with robust security features and access controls, allowing organizations to secure their data more effectively. Traditional methods like emailing spreadsheets or using FTP can lead to vulnerabilities and compliance issues.

Integration and automation capabilities

Many modern cloud platforms integrate with various tools and applications that organizations use. This can help you automate data processes, increasing efficiency and reducing the likelihood of errors compared to the manual operations of traditional approaches.

The cloud data platform should manage the complexity of copying data by moving it efficiently, increasing performance, and lowering overall costs. The application development team can grant access to specific objects in the database, utilizing the same access controls discussed in Chapter 1. This ensures a consistent approach to managing security for the data assets.

Choosing Where to Process the Data

As an application builder, you have choices about where to process the data. You can perform the processing either within your own environment or in the customer’s environment.

With the managed application model discussed in Chapter 1, you can utilize secure data sharing to give the consumer of the application access to do their own research on a subset of the managed data. Your application can perform additional processing, enhance and enrich the data, and share it back to your customers without the need for cumbersome file transfers or API development, as shown in Figure 3-2.

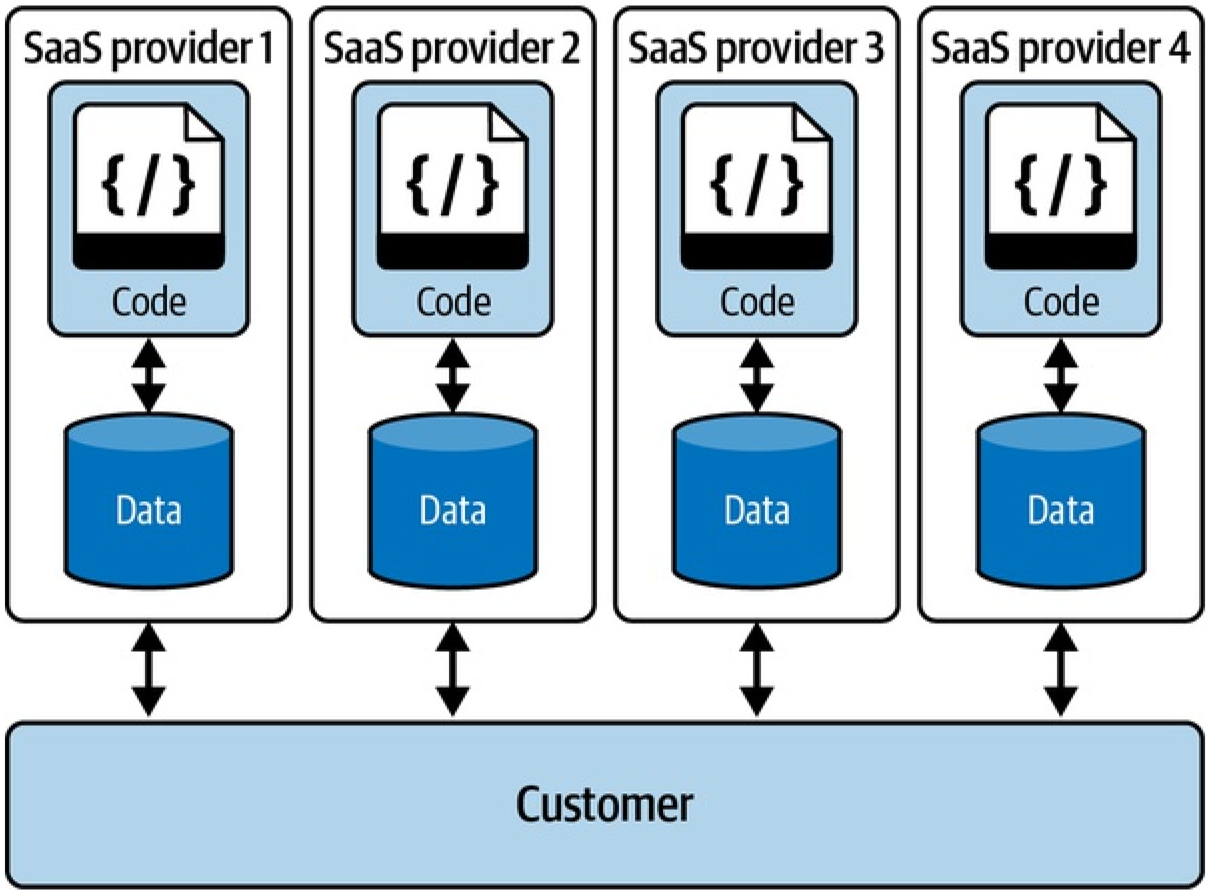

Figure 3-2. In the traditional managed application model, each SaaS provider holds the customer data separately.

App providers who choose this model are able to protect their code, because processing happens within their provider account, not the consumer's account. This enables you to fine-tune performance and scalability, ensuring that your customers receive a level of service that meets their expectations. Customers may be interested in this model for other reasons as well:

- They might not have all the data required within their own environment to generate the required insights and therefore need to rely on an app provider to take care of this for them.

- They may not have sensitive data and thus require less control and rigor around the handling of their data.
- They may have a clearer idea of the costs than if they process data within their own environment.

Alternatively, you may combine your processing with your customers’ cloud data platform and leverage their compute and data, following the connected application model (see Chapter 1). With this approach, your customers can keep their data within their own environment (see Figure 3-3), allowing them to consistently enforce their data governance policies and avoid unnecessary exposure.

For application builders, this model especially appeals to customers who require control over their data or need to ensure their data remains within a specific cloud region. By taking the processing to the customers’ data in the connected application model, you can alleviate the concerns related to compliance and privacy that often arise when sharing data with a third-party application provider. The downside is that some of your intellectual property (IP) may be exposed.

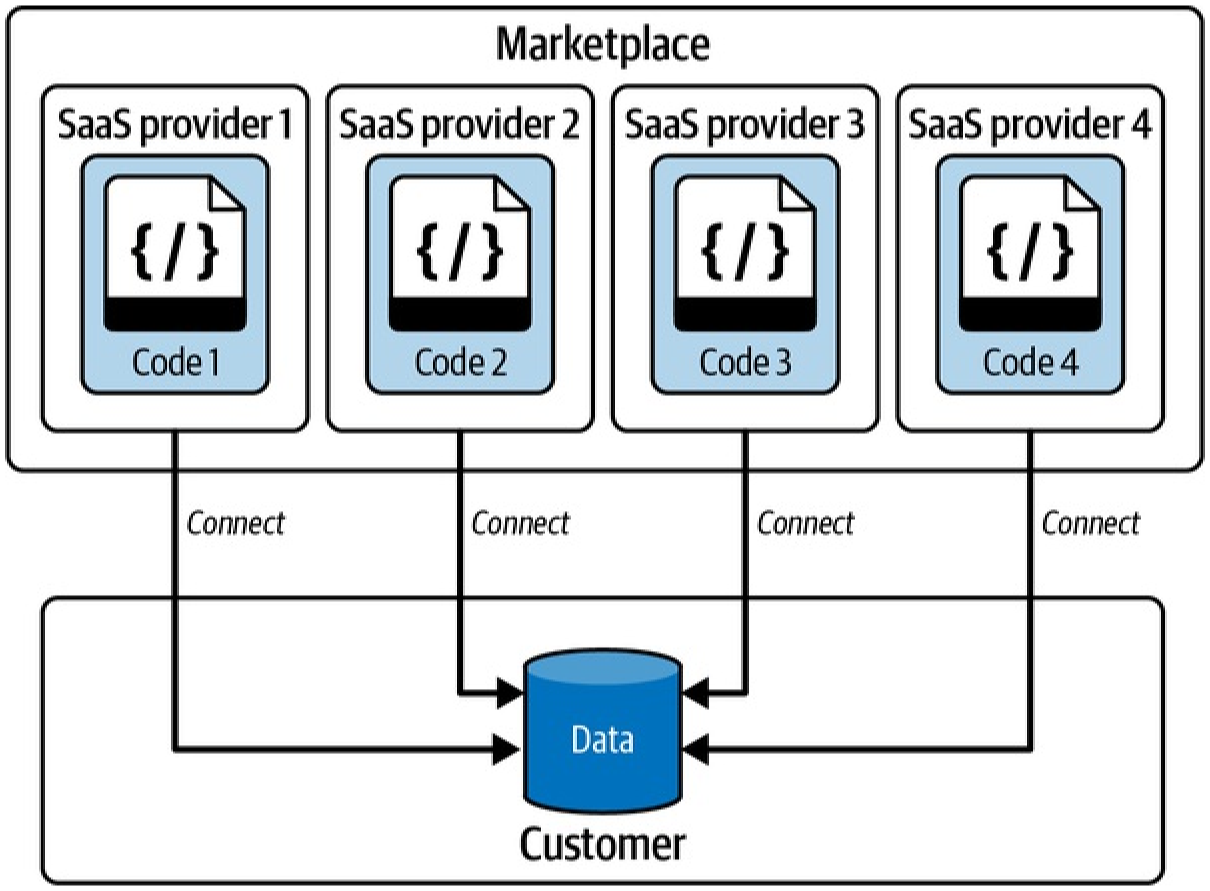

Figure 3-3. The connected model approach allows customers to retain ownership of their data while providers run their application code on the customers' data without movement.

Next, we will examine the role of the marketplace in revolutionizing application distribution and the nuances of the rapidly evolving data marketplace. Fostering closer ties between application builders and customers not only simplifies transactions but also aids in the discovery, purchase, and sale of data services online. It can also help the application builder protect all of their IP, if done correctly.

Modernizing App Distribution

A data marketplace is exactly what it sounds like: an online marketplace for data. It’s a central platform where your organization can access, share, and even monetize data assets. Think of it as a digital hub where data from various sources, both internal and external, can be easily discovered and utilized. This can include structured data like sales figures or customer information, as well as unstructured data like text or images.

Using a marketplace to distribute your application is an excellent decision. Organizations will be able to discover your application based on their specific requirements and easily procure your services. For an application builder, a marketplace provides the necessary infrastructure to list and host your app. It also allows you to configure various pricing and licensing models and contracts to present to your customers before granting access to your application.

As an app builder, you can achieve several benefits by leveraging a marketplace for app distribution:

Extend your reach
: Shorten sales cycles and scale faster by offering your application to a broader target market.

Improve customer experience
: Provide secure, controlled access to applications and data, which is automatically updated no matter your customers’ region or cloud.

Speed to market
: Leverage flexible pricing plans, standardized contracts, and billing

infrastructure.

Manage app versions

Push the appropriate version to each customer in a protected area of the customer’s account.

Through the marketplace, your customers gain the ability to search for relevant applications based on industry or use case, as well as to purchase the applications, accept commercial terms, and install or upgrade applications without the need for direct communication with you. This streamlines the process, making it efficient and frictionless. Additionally, the marketplace provides valuable insights that allow you to understand the popularity of your application and evaluate customer and market behavior. These insights help you identify areas for improvement so you can enhance your application accordingly.

Moreover, a marketplace can support internal-facing services, offering applications that cater to various analytic use cases across business entities. These applications may include customer segmentation, data quality scorecards to track the accuracy and completeness of internal systems, and a data catalog that serves as an effective search engine for discovering enterprise data assets within the organization.

Summary

In the ever-evolving world of data applications, efficiency and user experience are paramount. These applications must be agile, swiftly adapting to fluctuating data sets and formats. Raw data offers immediacy, while conformed data brings standardization, making querying more efficient.

Minimizing data movement is key to achieving optimal efficiency. The less data shuttles between systems, the fewer costs and complexities are incurred. Transitioning from traditional ETL processes to cloud-enabled ELT methods streamlines operations and offers greater flexibility. Moreover, leveraging techniques like CDC can focus processing on just the data changes, ensuring data integrity and timeliness without excessive resource use.

Assessing the efficiency of data pipelines and evaluating trade-offs between raw and conformed data based on specific use cases are important considerations. Cost-benefit analysis is crucial, as more frequent data processing may not always be justified by the value of the insights obtained. These considerations all contribute to a thoughtful approach that is tailored to individual business needs.

Modern application distribution is experiencing a revolution with the emergence of digital marketplaces. For an application builder, marketplaces offer a platform to reach a wider audience, provide a richer user experience, accelerate product launches, and adeptly manage app versions. Customers can discover, purchase, and manage apps with minimal hassle. Additionally, insights drawn from marketplace interactions can guide enhancements and innovations. Beyond the external marketplace, internally tailored services cater to diverse analytic needs, streamlining processes like customer segmentation, data quality tracking, and enterprise-wide data discovery.

Conclusion

Data applications generate tangible insights through the efficient processing of large volumes of complex, rapidly changing data in a variety of formats. The latest evolution of cloud data platforms provides the advanced capabilities necessary for application builders to exploit data to its full potential and offer innovative applications to their customers. The flexibility to adapt to data source changes and the ability to swiftly deliver insights to end users within a scalable computing framework set modern cloud data platforms apart.

Unlike their traditional counterparts, modern cloud data platforms provide the agility required for application development to thrive. In this report, we've covered the main considerations for selecting a cloud data platform to support your application development efforts. These best practices will help unburden your developmental teams from the complexities of data handling, allowing them to concentrate on building successful, customer-centric applications. Key features include:

Elastic scalability
A cloud data platform leverages the benefits of cloud technology by offering virtually limitless storage and computing resources. The platform caters to the demands of diverse tenants and workloads while ensuring elasticity, not only meeting SLAs during high-demand periods but also

maintaining cost-efficiency during times of reduced demand.

Cross-cloud deployment

This allows your applications to run seamlessly across different cloud providers or platforms, enhancing flexibility and reliability and potentially lowering costs.

Extensibility

This eases the process of adding new features or making changes to applications, facilitating continuous improvement and adaptation to changing needs. It empowers developers to extend the platform’s native functionality using a range of tools that enable encapsulation of business logic, access to external APIs such as machine learning models, and creation of interactive apps using familiar programming languages. This promotes an Agile approach to data manipulation, analysis, and application development, ultimately speeding up time to market.

Security

Security for data-intensive applications is multifaceted, encompassing robust measures such as encryption, authentication, RBAC, and fine-grained security controls. Combining these features with data encryption safeguards data at rest and in transit, forming a comprehensive security framework. We should standardize this framework to prevent constant reconfiguration, ensuring the protection and governance of sensitive or regulated data during application deployment.

Tenancy options

Different tenancy options (single and multi) offer various levels of isolation and resource sharing, allowing your teams the flexibility to tailor your application offering to the specific needs, performance requirements, or budget constraints of your customers.

Workload isolation

Ensuring customer workloads will not impact one another is critical in a multi-tenant environment. In addition, workload isolation prevents the different processes within a tenant’s environment from competing for resources. This allows data pipelines to run while providing application services without performance degradation.

Separation of storage and compute

Separating storage from computing resources allows for independent scaling and management, improving data processing efficiency and flexibility within the cloud.

Efficient data processing

Quickly and accurately handling large volumes of data is a crucial aspect of how cloud data platforms allow timely decision making and responsiveness in application development.

Application distribution

Leveraging features such as live data sharing and marketplaces helps you market, monetize, and reach global markets at scale. The benefits of immediate access and elimination of storage costs make live data sharing a must for modern data platforms.

In conclusion, modern cloud data platforms have redefined the application data stack, positioning organizations for unprecedented success and growth. This report has illuminated the essential aspects to consider when choosing a cloud data platform, equipping you with the insights needed to streamline application development processes and drive customer satisfaction. Embracing these platforms is a strategic step toward transforming your business with data and creating impactful applications that resonate with contemporary market demands.

About the Authors

Adam Morton is an experienced data leader and author in the field of data and analytics with a passion for delivering tangible business value. Adam’s continued commitment to the data and analytics community has seen him formally recognized as an international leader in his field when he was awarded a Global Talent Visa by the Australian Government in 2019. Today, Adam works in partnership with Intelligen Group, a Snowflake pureplay data and analytics consultancy based in Sydney, Australia. He is dedicated to helping his clients overcome challenges with data while extracting the most value from their data and analytics implementations.

With over 20 years of experience in engineering and leadership, **Brad Culberson** has designed and built countless applications leveraging huge datasets at high concurrency. Brad is recognized across the industry as an expert in columnar/kv/oltp datastores and developing highly scalable/efficient applications.

Kevin McGinley currently leads the Customer Acceleration Team (SnowCAT) at Snowflake, which focuses on accelerating customers' adoption of Snowflake’s most strategic and bleeding-edge features, especially for data applications. He has over 25 years of experience building high-scale analytic applications across a variety of industries and company profiles.