The 3-legged stool

Venn Diagramming Vendia

How Vendia Share remakes the building blocks of modern IT infrastructure into secure, compliant real-time data connectivity and sharing solutions

Vendia

Table of contents

Venn Diagramming Vendia

Why blockchains, databases, and APIs can't cut it on their own

Who should read this

- CIOs, CTOs, CDOs, and cloud architects who need to store, share, or operate operational data that spans applications, companies, departments, or clouds
- Business leaders who need insight into why databases, blockchains, and APIs each lack critical features needed to accomplish these tasks…and why DIY projects to combine them using cloud "Lego pieces" frequently result in cost and time overruns
- Developers and technical leaders interested in how Vendia's API-enabled data platform can replace the mundane work of setting up and operating APIs, databases, integration hubs, and other cloud infrastructure with a model-driven platform that offers SaaS-style simplicity and out-of-the-box scalability, compliance, security, and monitoring capabilities
- All readers looking to avoid the cost and delivery challenges of bespoke, point-to-point data integration projects with operational data

Executive summary

1. APIs are commonly used to "bridge gaps," enabling critical business data to be shipped between companies, organizations, applications, clouds, geographies, or across other IT/infosec boundaries. However, despite progress in API tooling , APIs remain "ignorant pipes." Without massive, continuous API polling activity, they leave the parties involved with data that becomes inconsistent, incorrect, and out of date.

2. Operational databases offer low latency, high-throughput, and highly available solutions for application storage and transaction processing (at least in the cloud). But, by their nature, they are owned and operated by a single entity. This makes it impossible to materialize their content in another company's IT infrastructure in a way that's guaranteed to be accurate without massive custom investment on both company entities. Despite knowing the customer's data model in detail, databases offer no help when it comes to crafting model-centric APIs or other elements of a data sharing stack.

Executive summary (cont.)

3. Blockchains achieve an ordered, consistent representation of data that can span clouds, companies, geographies, and IT stack technology choices. But they're one of the least successful enterprise adoption stories in the history of IT due to their high latency, lack of support for files, limited "one box" deployment model, and disregard for cloud integration.. In addition, their lack of a data model requires application developers to spend months or even years developing object-relational mapping (ORM) and other extensions (e.g., file support and transactions) on top of a blockchain's primitive key/value store in order to recreate the most basic "out of the box" features that a cloud database offers.

4. Vendia combines all three technologies (APIs, databases, and blockchains) to create a solution that solves the limitations of each. Starting from a customer-provided data model, Vendia's solution generates a custom and customer data model-centric API, a scalable and compliant cloud storage tier, and a full spectrum of consistent data sharing and access controls that work across different applications, companies, clouds, and geographies. And delivers it all with SaaS simplicity and zero infrastructure footprint to own or manage. Vendia relieves developers from having to piece together all the elements of cloud infrastructure, on different clouds, in order to build applications and data platforms that can effortlessly span those gaps.

5. Customers use Vendia's operational data and API platform to quickly create a variety of secure, fault-tolerant solutions: Ticketing systems, mortgage and insurance settlement, supply chain and logistics tracking, business contract tracking, and much more. Compared to manually building point-to-point solutions using bespoke code and self-managed infrastructure, Vendia offers up to a much faster time to market, lowers ongoing operational and maintenance costs, and lowers development costs.

Introduction

Enterprises and developers have been on a decades-long quest to simplify the challenge (and lower the cost and risk) associated with building IT solutions. In recent years, IT heavy lifts have grown even heavier. Lift is driven by the growth in mobile and web apps on the front end and the growth of APIs and automation on the back end. Combined, these trend lines have had two major impacts:

1. More operational data being produced and consumed with every passing day
2. Data is fleeing central IT's control—migrating into SaaS applications, public clouds, department-level data stores, and upstream and downstream business partners

Despite a seemingly endless supply of open source libraries and a host of fully managed cloud services for every conceivable need, these trends continue to keep many fundamental IT goals out of reach.

Polycloud is failure-prone

Avoiding cloud lock-in (while also achieving high availability) requires data platforms, APIs, and application states that can span two or more clouds. And yet building multi-cloud applications or data sharing solutions is one of the most costly, expensive, and failure-prone challenges facing companies today.

Partner data sharing is rife with challenges

Sharing data with partners (e.g., suppliers, logistics, and manufacturing as well as loyalty program and co-marketing/alliance partners) is a fundamental and critical to business success. Yet achieving consistent, trustworthy shared data when it spans different companies and IT stacks remains one of the most challenging, costly, and security-concerning aspects of any company's IT portfolio.

DIY-ing it doesn't scale and is expensive

Modeling use cases and business-specific data, whether it's a formal model-driven development program or an informal approach, requires that developers piece together a wide variety of technologies and capabilities, from databases to file storage solutions to business logic and workflow hosting platforms, to hosted API solutions. Before being production-ready for either its owner or the owner's business partners to consume, each of these bespoke solutions must be vetted for security, scalability, availability, compliance, and more. On top of that, the solution must be monitored, operated, upgraded, and secured in perpetuity… including supporting the evolution of the data model as the business needs or business partnerships change. **The costs of recreating this pattern over and over again, especially for point-to-point APIs between partners, are staggering.**

Companies have tried various approaches to lower these costs, reduce risks, and improve time to market for IT projects. Creating solutions that can share real-time data across applications, companies, clouds, geographies, and IT stacks is both difficult and costly and, at the same time, incredibly repetitive.

What if there were a better way to do it?

In the following sections, we'll explore why that's true and how customers can experience lower costs, faster time to market, and achieve powerful outcomes, like multi-cloud, without the need for complex or time-consuming coding. First, let's explore why APIs, databases, and blockchains don't cut it as standalone solutions, and then use those identified gaps to develop a set of requirements for an operational data sharing platform.

APIs: Good at moving bits around, bad at storage and consistency

APIs are the central building blocks of services, especially modern, cloud-based designs. They encapsulate complexity and provide operational and implementation isolation between different parties. Building great APIs should be simple: all major cloud providers offer highly scalable, fully managed services that remove virtually all infrastructure challenges from both API owners and consumers. Google Apigee, Amazon API Gateway, and Azure API Management handle the challenges of hosting, deploying, scaling, and making API infrastructure fault tolerant. But despite this, building systems that share data across organizations or business partners using APIs remains one of the most challenging and expensive propositions an IT team faces. **Why? Because APIs are not-aware pipes; they're ignorant of the data they carry or the systems they connect.**

Semantically, an API is no different than email. Like an email service, an API doesn't understand the content of the payloads it delivers; the API's job is just to get payload contents to where they're supposed to go. If any API was last called 24 hours ago and the data it carried then is no longer accurate, no one notices.

The situation is even worse when multiple parties are trying to communicate over APIs. Just building that "N-squared" communication framework alone is a huge lift, and then on top of that it doesn't even work well. There's no guarantee that any of the parties have a view of reality consistent with any of their peers. That lack of guarantee makes building virtually anything, from supply chain modeling to financial settlement to contract enforcement, somewhere between radically complicated and outright impossible.

"Fixing" an API to do better would require several things to happen simultaneously. The API would need to start modeling:

- The data schema
- The parties and their relationships (including access controls)
- The state of data replication among the parties to create transactional ("ACID") consistency

If your reaction is, "But that sounds like a database or a blockchain!", you're not wrong.

So let's look at those technologies and see how they fare in isolation.

Databases: Good at storing state, terrible at sharing it

Databases, particularly cloud ones, are good at storing scalar information. Modern SQL and NoSQL cloud databases offer unlimited storage, massive scalability, four or more "9's" of availability, and transactional semantics. They are also good at understanding data models, also known as schemas, and some of them even allow those schemas to evolve over time. This makes databases the ideal solution as the operational storage tier of an application.

Unfortunately, modern cloud databases are also designed to serve exactly one customer (account) and to only work on a **single** cloud.

That makes solving some of the most important challenges a business faces incredibly difficult (e.g., avoiding cloud lock-in and sharing data with departments or partners who may be on a different cloud). Even though the data is stored accurately, getting it consistently, correctly, and completely replicated into the database of some other party is a huge lift; it exacts a vast amount of time, risk, and cost on development teams. The irony is that those costs and risks are also highly repetitive. Sharing data about a car chassis, a bottle of ketchup, or an airline loyalty purchase is essentially identical at the infrastructure level.

"Fixing" a database to do better would require several things to happen simultaneously. The database would need to:

- Offer a way to expose and exchange data with other parties through a secure API
- Operate transparently on different clouds
- Model the parties and their relationships (including access controls)
- Model the state of data replication among the parties

If that gap sounds like some kind of mashup of APIs and blockchains, you're right!!

Next, let's see how blockchains fare on their own.

Blockchains: Good at multi-party consistency, terrible at everything else

A relatively recent technology introduction, blockchains achieve an interesting outcome: **They create a consistent data representation among multiple parties without any one party needing to be in control.** By their nature, most blockchains also offer a built-in ledger (making application logs unnecessary) and they offer the ability "time travel" (retrieving historical versions of the data objects being stored).

When first introduced, blockchain technology was heralded as an IT savior: Finally, there was a solution to sharing complex application state across clouds, companies, geographies, and technology stacks that could offload one of IT's biggest challenges.

So, what went wrong? Unfortunately, while blockchains do offer a solution to the low-level challenge of creating multi-party data consistency, they failed virtually every other aspect of enterprise-grade IT.

1. **Blockchain protocols ignore the public cloud**
   That's problematic, because most of the data being produced and consumed is being done so in the public cloud; a technology that actively tries to avoid cloud integration is essentially dead on arrival.
2. **Blockchains fail to scale**
   One of the downsides of not availing themselves of public cloud services is that blockchains have a hard time scaling. Their nearly universal "one box deployment" model means that, once they reach the limits of what a single server can achieve, it is game over. That makes it pretty obvious that neither private nor public blockchains will ever achieve throughput rates needed by real-world businesses.
3. **Blockchains forgot about fault tolerance**
   Blockchain architects confused the difference between decentralization and fault tolerance or high availability. To build a highly available solution, customers need to deploy multiple blockchain nodes, making the already complex

Databases have been around for half a century, and have accrued a wealth of knowledge about what applications and users require. Unfortunately, blockchain inventors ignored all those lessons and skipped the basics, including data models, data model evolution, primary keys, joins, flexible query languages, ACID transactions, and more. Would-be enterprise adopters of blockchain technology inevitably find themselves saddled with the challenge of trying to recreate database semantics and customer-centric data APIs on top of it.

> **"Multi-party consensus is a powerful technology, but ultimately, it's only a building block. Just like databases and APIs.**

Databases, APIs, and blockchains are necessary …but not sufficient as standalone solutions

It seems we've reached an impasse: We need the benefits of cloud databases AND managed APIs AND modern blockchains, but no one of these technologies, by itself, solves the underlying challenges that companies face today (see Figure 1).

None of these approaches in isolation provide a complete data sharing solution capable of modeling data accurately while bridging company and cloud divides and scaling to enterprise-grade levels of throughput.

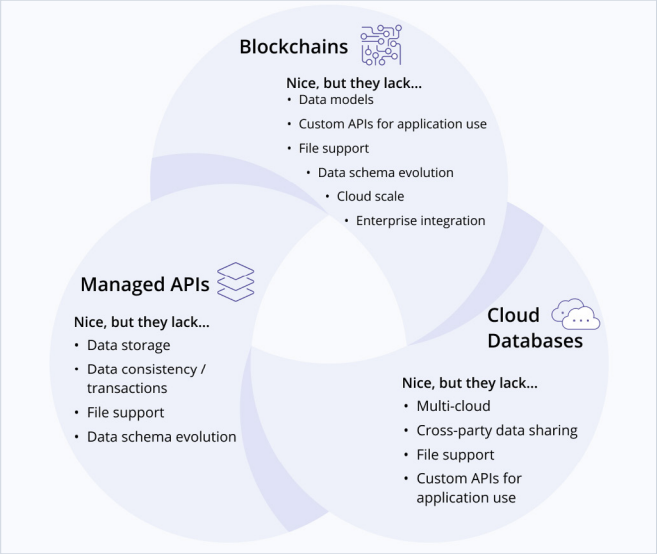

Figure 1: APIs, databases, and blockchains each bring something to the table, but no one technology suffices on its own.

Other approaches such as real-time streaming (e.g., Kafka, Cassandra, Amazon Kinesis) solve elements of the challenge, particularly for single owner-solutions. But none of the purveyors of these technologies have been able to expand beyond their narrow boundaries, limiting their ability to help teams create broadly suitable solutions (see Figure 2).

Data sharing approach	Keeps data consistent?	Allows other accounts, clouds, organizations?	High scale & low latency?	Long-term data storage?
Conventional APIs	✗ No	✓ Yes	✓ Yes	! Depends on implementation
Central databases	✓ Yes	✗ No	✓ Yes	✓ Yes
Events / Streaming	! Depends on service(s)	✗ No	! Depends on service(s)	✗ No
Legacy Blockchains	✓ Yes	✓ Yes	✗ Neither	✓ Yes
Vendia	✓ Yes	✓ Yes	✓ Yes	✓ Yes

Figure 2: Compare and contrast data sharing approaches and technologies

Looking at the gaps in these individual technologies, a set of requirements naturally emerges that defines what a powerful, easy-to-use data sharing platform needs to be "enterprise-grade.

In the next section, we look at how Vendia combines these technologies to create a platform that offers the benefits of each constituent technology without compromising and then explore how customers can adopt it for a variety of use cases.

Vendia: Smart, “data-aware” APIs across parties and clouds

Exploring the limitations of APIs, databases, and blockchains exposed the critical set of needs an enterprise has for operational data.

An ideal real-time data-sharing solution simultaneously needs to offer

- The strong operational, security, and systems isolation of a conventional API
- The security, unlimited long-term data storage, and easy queryability of a conventional centralized database
- Similar storage capabilities for files
- The cost efficiency and scalability of an event-based architecture
- The “time travel,” guaranteed delivery, and strong ordering guarantees of a real-time streaming solution
- A blockchain’s ability to easily span clouds, companies, and geographies with a consistent data representation

Vendia’s approach combines the strengths of different approaches to overcome the limitations of each (see Figure 3).

Vendia combines API, database, and blockchain technology—all three, reimagined as a single, powerful, cloud-native platform. Together, as Vendia Share, they can do what no one single technodlogy is capable of in isolation.

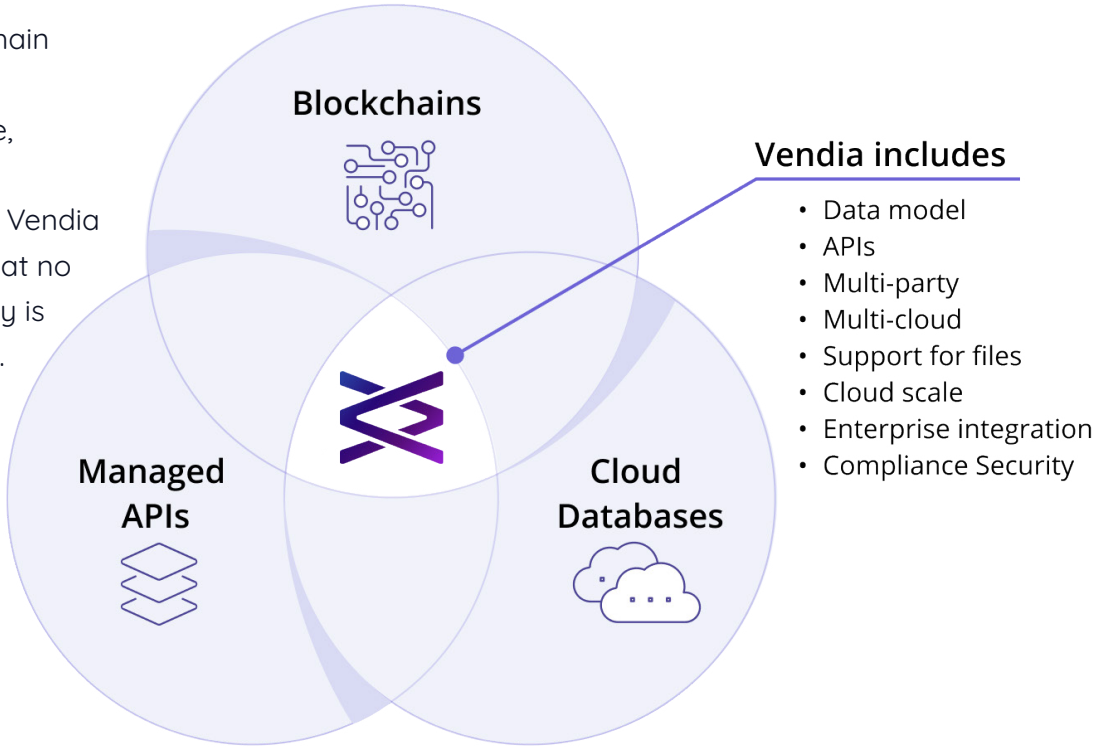

Figure 3: Figure 3: Vendia as a venn diagram, combining the best attributes of APIs, databases, and blockchains in a unified SaaS platform.

Vendia Share makes building data-sharing platforms and applications fast, simple, and secure by design:

1. Each participant has a node containing one or more cloud-hosted databases and filestores, capable of 11 9's of durability storage with unlimited capacity.
2. On top of that, a high speed data replication and consensus mechanism emulates a blockchain's ability to create consistent replicas; the mechanism does so with massive parallelization and cloud-enabled scale and throughput.
3. A user-provided data model is used to automatically generate modern GraphQL APIs, including subscriptions that offer real-time streaming and they're all designed around the application's data schema. These capabilities include versioning, "time travel," auditing, and lineage analysis—and they're available without needing to write even a single line of code. Conventional and familiar database concepts work as expected (e.g., transactions), and files are handled as a built-in feature with full ACID support, just like other data types.
4. All services are delivered in SaaS style with zero infrastructure footprint to manage or maintain; fault tolerance is built in (instead of left as an exercise to the customer, as in conventional blockchains).
5. Everything is built using scalable and fault tolerant serverless cloud technology, allowing tight cost enveloping, environmental friendliness/a low carbon footprint, and automatic scalability on a per-request basis.

Figure 4 illustrates Vendia's architecture, combining the best of managed APIs, scalable cloud databases, and enterprise-grade blockchain technology.

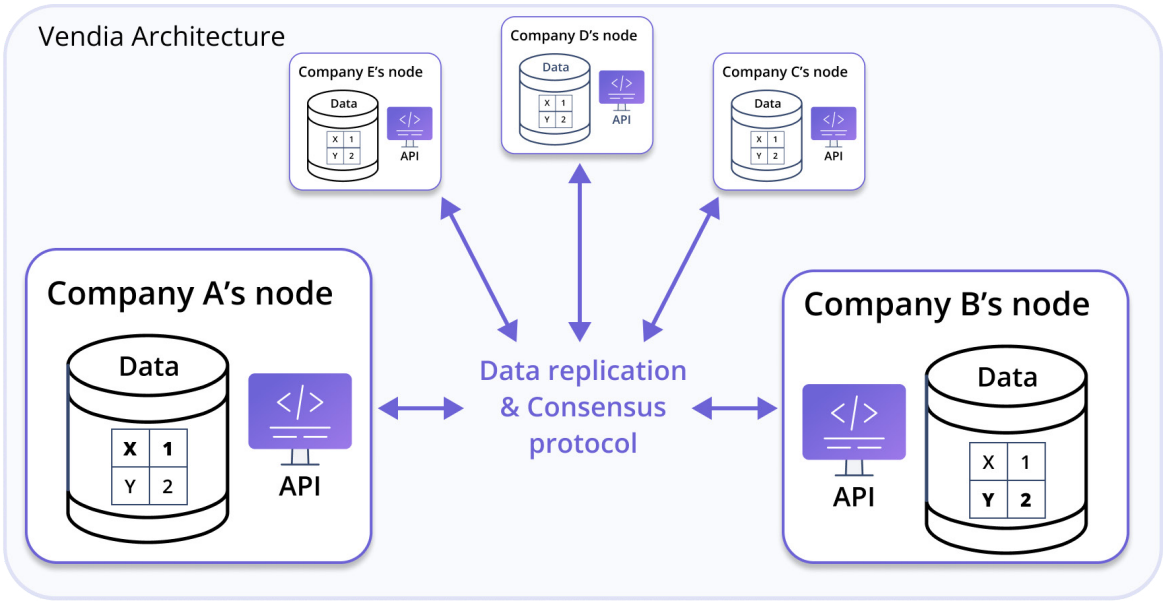

Figure 4: Vendia's "three-legged stool" architecture. An operationally and security isolated Node is created in the cloud for each party, including file and scalar data storage. Vendia transparently and automatically synchronizes the data among all nodes using patented blockchain and serverless algorithms. A powerful GraphQL API is generated automatically from (and permanently synchronized with) the application's data model, including real-time streaming support that keeps each node's applications consistent with one another. Despite its power, the "no code" approach is simple, with zero infrastructure footprint to manage.

Simplicity by design

Combining technologies can sometimes lead to systems that are more difficult to use, especially if their consumers are required to understand the complexity and details of all the constituent pieces. Fortunately, Vendia's use of a data model compiler actually reduces the set of things a developer needs to learn relative to using APIs, databases, or blockchains in isolation, allowing the complexity in these low-level technologies to stay behind an abstraction barrier as "implementation details." (See Figure 5)

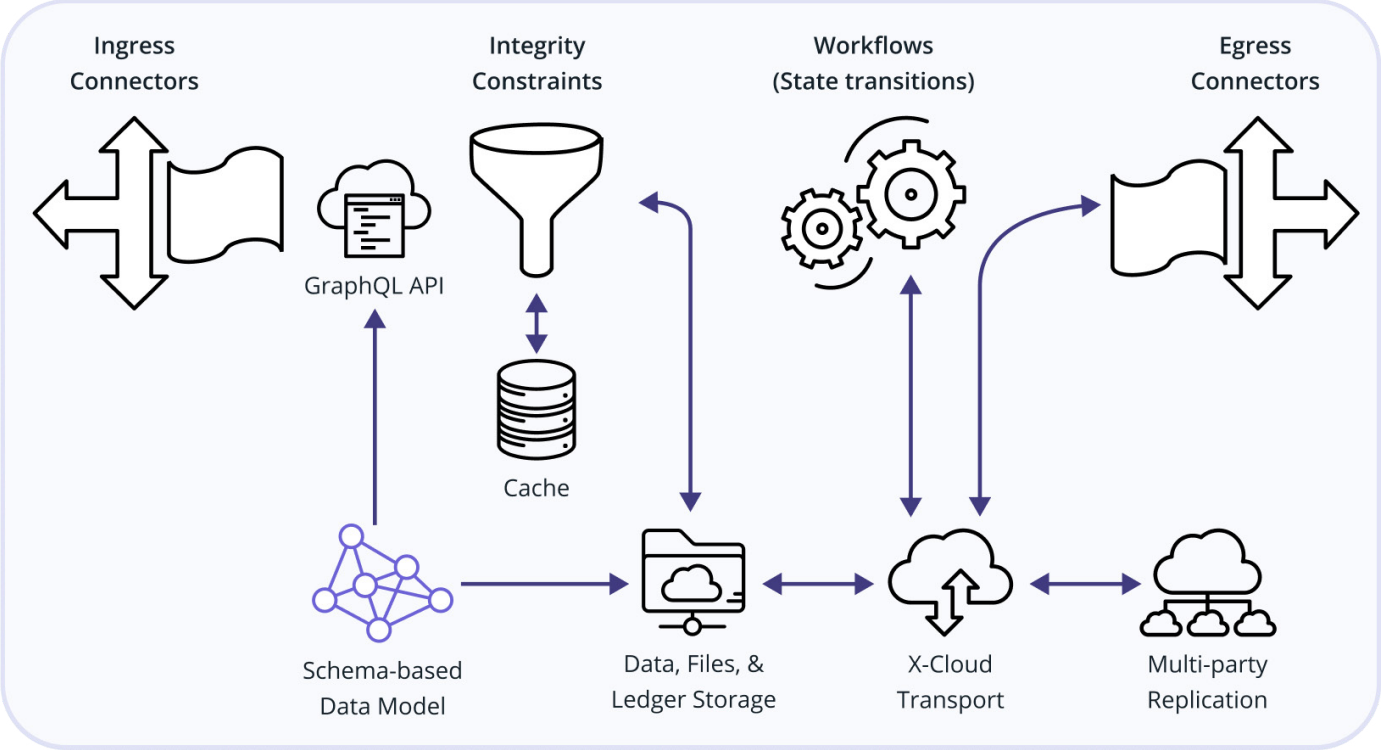

Figure 5: The data model is used to generate the (1) data storage, (2) file storage, (3) APIs, (4) data-model aware consens and replication layer

As in a conventional database, the customer brings a data model (a description of what the "tables and columns" should look like).

Vendia's data model compiler turns this description into several outputs:

1. A customer-specific storage tier, capable of holding both business and file data
2. A consensus protocol that handles data replication across parties and clouds, maintaining consistency and guaranteeing ACID transaction semantics
3. A modern, GraphQL API that makes it easy to build create, read, updated, deleted (CRUD) application logic that's generated from the data model without the need to hand craft

Developers using Vendia Share don't have to concern themselves with how these elements interact, nor do they have to worry about complex infrastructure hosting or deployment. All the elements above are delivered in SaaS fashion, fully managed and cloud-hosted.

Vendia's sharing topology

In addition to the data model itself, there is one another input to the Vendia platform: The sharing topology. Vendia Share makes it easy to connect to business partners and their data, even when they're on different clouds. That means each party can independently control which public cloud (e.g., AWS or Azure) and region (e.g., us-east-1 or us-west-2 on AWS) they want to reside in. A single party can even deploy multiple nodes, making it easy to create geo-separated, multi-region redundant systems with as little as two lines of configuration.

Additional parties can be invited by sending a secure invitation over email. Vendia handles the heavy lift in the background, automatically creating and deploying the new party's cloud infrastructure, then syncing its data to reflect the correct subset of data from historical transactions—all without any of the parties needing to write a single line of code.

Finally, Vendia Share makes it easy to configure individual nodes. Connectors to popular data sources (e.g, Salesforce), data ingress and egress to cloud queuing systems, and smart contracts expressed as cloud functions written in any language are all easy to configure "on chain" through simple configuration settings from the API, CLI, or developer UI.

These configuration-driven mechanisms allow developers to gain powerful ROI with fast time to market.

> **"Given a data model, a production-grade, multi-cloud, multi-region, multi-party solution can be deployed and connected in under 10 minutes by one developer.**

Contrast that with typical blockchain and public API deployment cycles that are usually 12-24 months of time with 10-20 developers required **per party**.

In the next page, we'll take a look at a sample use case in action.

A customer example: Co-marketing programs

Let's take a simple but common example of data sharing, co-marketing programs between two (or more) companies or divisions. This can happen when companies want to discover common customers or leads in order to create joint marketing campaigns, loyalty program networks, or similarly overlapping outcomes. For two or more divisions within the same company, it can represent situations such as multiple Salesforce or other CRM deployments, where information is siloed and needs to be converged such as after an acquisition.

While sharing this information is critical to a successful outcome, it's also frequently just as important to avoid oversharing. Regulatory requirements, compliance rules such as general data protection regulation (GDPR) or California Consumer Privacy Act (CCPA), or the simple desire to avoid sharing information about unrelated customers means that the underlying platform needs to make it easy to do all of the following:

1. **Third-party escrow**
   Ensure the analyses, such as name and/or address matching, are performed in a secure escrow location (also known as a data enclave), rather than by the parties themselves. This avoids one party having to expose all their data to the other in order to perform comparisons or other data analyses or transformations in a conventional, centralized database owned by just one of the parties.

2. **Access controls**
   Each party needs an easy way to control what data (i.e.,columns and rows) is exchanged with the other party (usually via the escrow/data vault). This is required to ensure that each party retains control over which marketing and customer data is seen by the other party or by any shared analysis. This is also where inconsistent data served up by a conventional API could be dangerous: With compliance regulations such as GDPR in Europe and CCPA in California, getting a customer's personally identifiable information (PII) sharing settings wrong (through outdated or inconsistent values) could literally mean breaking the law.

3. **Multi-cloud, multi-region, and multi-CRM**
   While each party may be happen to operate their portion of this shared workload on a single cloud, it's often the case that they will each have made differing IT adoption and vendor decisions over time—different clouds, different geographic data center regions, different CRM vendors, etc. The sharing solution cannot assume everyone has migrated to a common vendor and cloud as an a priori requirement, or no progress can be made. Even within a single partner, customer information reside in CRMs, custom application databases, and other locations.

To keep things simple, we'll assume there are only two parties and that matching (determining if two customer records are the same person) is based on a customer's last name and phone number.

The matching logic is simple: If these two fields are equal, then the remainder of the information in each of the matching records is shared with the other party, unless it's marked as private. The tables below show the data fields for each of our two imagined companies, Acme Corp and CoolTech, Inc.

CoolTech Customer Info Field	Type	Share	Must match
First Name	String	Yes	No
Last Name	String	Yes	Yes
Phone	10 digit number	Yes	Yes
Addess	String	Yes	Yes
Number of downloads	*Number*	*No*	*No*

Acme's Customer Info Field	Type	Share	Must match
First Name	String	Yes	No
Last Name	String	Yes	Yes
Phone	10 digit number	Yes	Yes
Addess	String	Yes	Yes
Last year sales	*Number*	*No*	*No*

Note that each of the parties has a field ("Last year sales" for Acme and "Number of downloads" for CoolTech) that they wish to remain private, even when a record is found to match.

For fields that don't match, the resulting record needs to preserve the source of the original information (since the values could differ):

Matched Customer Info Field
Shared Data (Last Name, Phone Number)
Acme Data (First Name, Address)
CoolTech Data (First Name, Address)

Creating this solution on Vendia Share requires uploading a data model, then creating a smart contract to perform the name comparisons and create the matching records. This can be accomplished easily, in under an hour, resulting in a fully scaled, production-grade deployment.

After that, each party can load their data, and then continue to add data over time. Vendia will continuously process records when either party adds them, automatically scaling up or down to handle varying workloads (see Figure 6).

These systems are all immediately production grade—they come with prebuilt scaling, fault tolerance, and both on-the-fly and at-rest encryption. Each company can set up custom authentication and authorization for as many systems as they like, even using additional nodes if they want their data to span multiple public clouds or regions for additional data backup or fault tolerance.

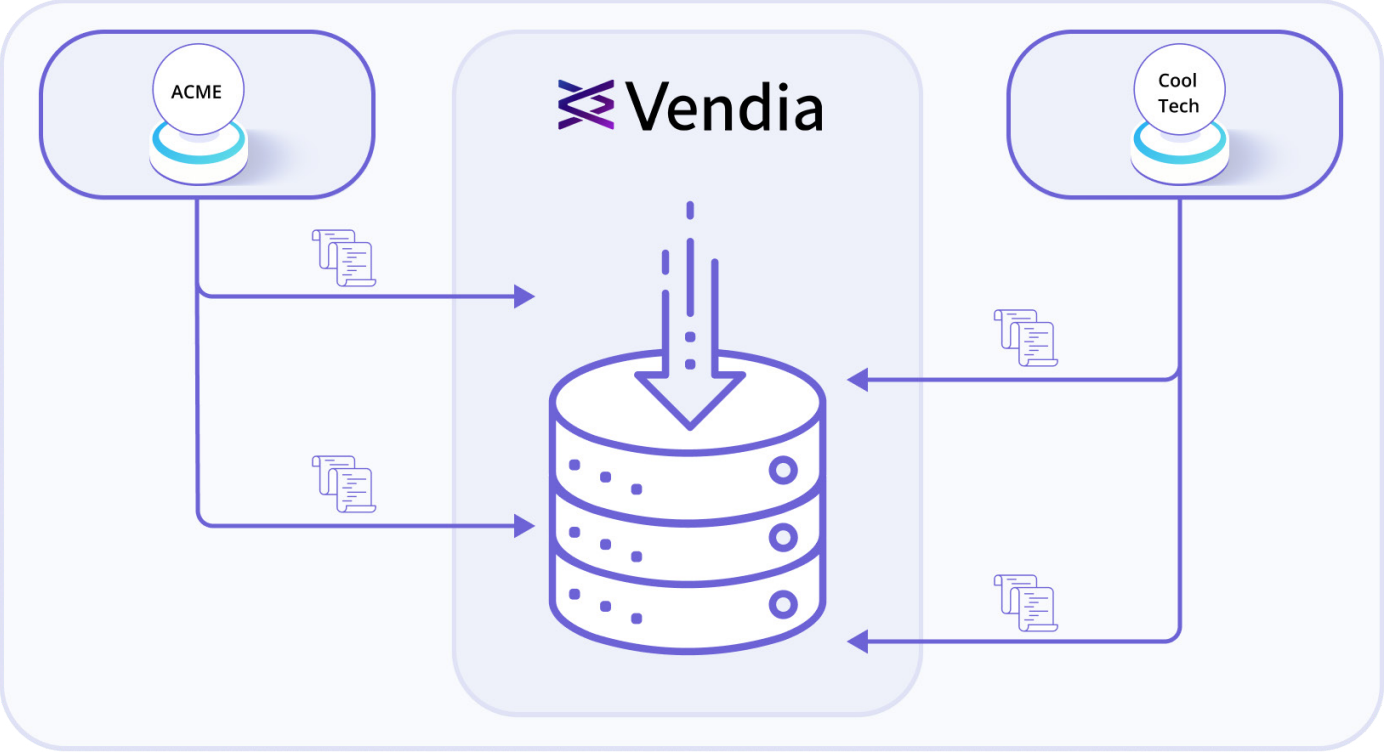

Figure 6: Acme Corp and CoolTech share records across time, accumulating a shared set of records and a real-time single source of truth.

Changes over time: Data model updates

Many application systems fail to recognize a key aspect of business: **The data needed to drive outcomes changes over time.**

One of the key strengths of Vendia Share is that it is designed not just as a one-time deploy solution, but as a system that supports companies as they adjust or add to the data they need over time.

For instance, In our example use case above, Acme Corp and CoolTech might decide to move beyond simple co-marketing and create a shared customer loyalty program. Now, they'll need to start tracking customer loyalty points and related information, too.

To do that, they need to modify their data schema to incorporate additional fields, such as loyalty point account tracking. If the data model were fixed, this would be impossible—they'd have to throw away their existing data sharing solution and start over! Fortunately, that's not the case with Vendia Share.

> **"Vendia Share's sophisticated schema compiler can understand schema additions and changes over time. It can incrementally modify the production-grade deployment to add new fields to existing data types as well as entirely new data types to the existing solution—without compromising the security or integrity of existing data and without requiring API clients to be rewritten.**

That latter point is especially crucial: If APIs have been shared with mobile or web clients, changing those APIs could break every app instantly. To avoid that, Vendia Share uses a sophisticated GraphQL API that is evolution-aware, and its schema compiler automatically detects and prevents any changes that could damage API clients.

The result? Changes are simple, seamless, and guaranteed to be safe for all clients.

That kind of guarantee is the holy grail of product and business owners (and the bane of development teams who frequently struggle to achieve it). With Vendia, it's impossible not to achieve that guarantee.

Changes over time: Partner updates

In addition to data model changes, there's another important way our use case example could evolve: The business relationships themselves change. For instance, in our simple example we had only two companies sharing data. Now, let's assume that four new partners decide to join this emerging loyalty program. Will that mean rewriting the entire application?

Fortunately, Vendia Share models business partnership sharing directly, making adding new parties as easy as sending them an email invitation. Vendia's secure data replication platform will automatically deploy the new participant's cloud resources automatically, with no coding required, into their cloud and geographic region of choice, then seamlessly replicate all required data to them. If a business partner should need to leave the loyalty program, removing them is equally simple, and they can take their accumulated data (though of course no data created post their departure) with them.

And while adding parties is most commonly thought of as a way to model business relationship changes, the same feature functionality can also be used by the existing parties for important IT outcomes. Additional nodes can be added to create multi-region backups, multi-cloud resilience, or additional operational boundaries or geographic scaling.

Conclusion

APIs, cloud databases, and blockchains are each an important building block of modern IT infrastructure but, on their own, they lack the capabilities to create real-time data connectivity and sharing solutions.

Vendia combines all three technologies in its platform Vendia Share, offering customers an easy, secure, and scalable way to create modern IT outcomes as simply as they create a new database table today.

With nothing more than a data model, Vendia customers can easily achieve a multi-party, multi-cloud, multi-region solution able to connect their applications and data with other divisions, other companies, and other clouds...all without having to write code or manage servers.

Vendia's customers use these capabilities to create secure, compliant solutions for loyalty programs, ticketing, financial settlements, file and media sharing, and other critical outcomes where accurate, real-time operational data is critical.

Learn more

- To learn more about how Vendia can make API development and management easier for data-driven applications, download our Smart APIs ebook.
- To explore modern application design that leverages modern data platforms, read about the Lean App movement on Vendia's blog.
- To explore customer use cases, visit Vendia's platform page.
- To see if your use case could benefit from a data-centric API approach or a data sharing platform, contact Vendia.

About the Author

Dr. Tim Wagner is the co-founder of Vendia and serves as CEO. Tim is the inventor of AWS Lambda and a former general manager of AWS Lambda and Amazon API Gateway services.

He has also served as VP of Engineering at Coinbase. Tim is the driving force behind the Lean Apps Movement, holds dozens of patents in serverless, cloud compute, and database technologies, and is a frequent keynote speaker at software conferences.

linkedin.com/in/timawagner

@timallenwagner

About Vendia

Vendia is the future of collective data intelligence, combining smart APIs, databases, and distributed ledger technology inside a single platform. Vendia's data automation cloud makes it easy to share data inside and outside of the organization in real time and with full visibility, governance, and control. Companies such as BMW, Delta Airlines, Resolution Life Insurance, and Fannie Mae use Vendia to automate contextual and compliant data flows between any-to-any systems for a harmonized, accurate view of data that unlocks speed, innovation, and cost savings. Learn more about us at Vendia.com and #UnchainYourData with Vendia.