WHITE PAPER

STATEFUL APPLICATIONS AT THE EDGE

yugabyteDB

Table of Contents

Introduction

Data is the lifeblood of the modern enterprise, driving new customer experiences and operating efficiencies. This data is now coming from everywhere - smart devices, mobile users, connected vehicles, building sensors, intelligent point of sale systems and more. Gartner predicts that by 2025, 75% of enterprise-generated data will be created and processed outside of a traditional centralized data center or cloud. Applications and users that consume the data and resulting analytics are also geographically distributed.

Edge computing is an emerging paradigm that distributes computation and data storage closer to where the data is being produced and consumed in order to reduce response times and save network costs. New edge applications exploit smart devices, mobile phones, and network gateways to perform tasks and provide services locally on behalf of the cloud—with the most powerful edge applications being stateful.

These stateful edge applications demand a new data architecture that takes into account the scale, latency, availability, and security needs of applications.

This whitepaper discusses new issues and challenges specific to stateful applications at the edge, and key considerations in designing a data architecture for edge applications. A new class of databases, distributed SQL, has emerged as a viable choice for the edge data layer, combining the best features of traditional RDBMSs and NoSQL databases for running transactional applications.

Defining Edge Tiers: From Cloud to Wearables

The goal of edge computing is to keep and use data close to where it is produced and consumed, integrating with core or more centralized applications that exist in the cloud. **But, where exactly is the edge?**

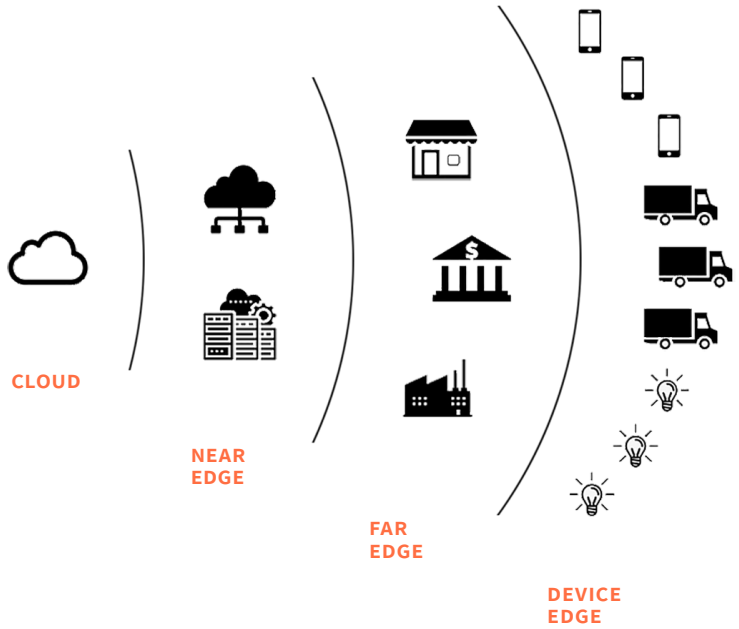


Figure 1: Different tiers in edge computing

Enterprises can deploy applications and databases across several tiers of infrastructure, each with its own unique set of properties. A typical enterprise service will consist of several applications, some of which run in a centralized cloud service and others outside it.

1. ***Device Edge:*** The device edge refers to mobile phones, IoT devices, wearables, and sensors in buildings or machinery. The device edge typically has extremely low compute and storage, so the databases running on them need to be lightweight. These databases are typically used for "store and forward" access. The device edge can often be disconnected from other tiers for extended periods of time, so applications and databases need to plan for offline operation and poor network connectivity.

2. ***Far Edge:*** The far edge consists of machines deployed near mobile base stations, inside shopping malls and retail locations, bank branches, or factories. The far edge sits close to devices and endpoints. Applications and databases running in the far edge are tied to the geographic location of deployment, and have low latency, high throughput access to data from devices. They have limited compute and storage resources available to them. Databases in the far edge are predominantly in private clouds, and need to be operationally simple to deploy and manage. The far edge can be disconnected from the near edge and cloud for extended periods of time.

3. ***Near Edge:*** The near edge refers to infrastructure that sits between the far edge and the cloud. This could be a public cloud region, a colocation site from a provider, a tier 2 cloud service, or a private datacenter. The near edge offers balanced compute and storage resources for applications. Databases deployed in the near edge are good for low latency access and data residency requirements, but need to contend with network partitions. Private and hybrid cloud deployments are the norm. Near edge applications and databases can also experience network partitions (though less frequently than in the far edge) and will need to plan for it.

4. ***Cloud:*** Cloud services such as AWS, GCP, and Microsoft Azure offer unlimited compute and storage resources, and access to cutting edge machine learning models. But, getting data from the edge to the cloud can incur high latency and be throughput constrained. Databases running in the cloud tier should be able to run in any public cloud. Enterprises are embracing a multi-cloud strategy to improve resilience, lower risk, and avoid lock-in. A multi-cloud managed DBaaS is ideal. Leading cloud services offer continuous availability (though recent infrastructure outages call this into question).

	DEVICE EDGE	FAR EDGE	NEAR EDGE	CLOUD
Description	Runs on the device, powers local data collection/ processing	Infrastructure closest to the device, farthest from the cloud	IaaS that typically sits between the far edge and cloud	Full blown IaaS. Usually the "brain" of the application
Infrastructure	Device – very low compute and storage *Examples:* cell phone apps, wearables, sensors in data centers/ buildings	Predominantly private cloud deployments *Examples:* machines deployed near mobile base stations, inside shopping malls or factories	Private and hybrid cloud deployments *Examples:* a public cloud region, coloca-tion provider (Equinix), tier 2 cloud, private datacenter	Public cloud *Examples:* AWS, GCP, Azure, etc
Latency	N/A	1-10 ms latency	5-50 ms latency	25-250 ms latency
Throughput	N/A	High throughput	Moderate throughput	Variable throughput/ high cost
Resources	Very low compute and storage	Limited compute and storage	Balanced compute and storage	Infinite compute and storage
Network Partition	Can get disconnected for extended periods	Can get disconnected for extended periods	Better resilience but can still get disconnected	Continuously available (though this is arguable considering frequent cloud outages)
Database Characteristics	Typically "store and forward" and support "access patterns on the device" Needs to be lightweight to ensure low resource consumption	Applications are tied to the location of deployment. Predominantly private cloud deployments. Needs to be operationally simple	Good for low latency and data residency needs Stack needs to be operationally simple	Has cutting edge machine learning models to leverage Ability to operate on any public cloud is important. Multi-cloud managed service is ideal

Note: This paper does not go further into the device edge since it is distinct from the other tiers.

Edge Applications in Different Verticals

Innovation at the edge is no longer the sole domain of a particular industry. Stateful edge applications are becoming prevalent across every vertical. Here are a few examples:

- ***Retail:*** Next generation point-of-sale systems, theft detection in store, onsite item lookups, employee identity management
- ***Banking:*** Mobile banking, customer identity management, fraud detection, retail banking services
- ***Automotive:*** Connected vehicles, autonomous vehicles, proactive maintenance
- ***Telecommunications:*** 5G applications, radio access networks, autonomous networks
- ***Industrial IoT and Manufacturing:*** Industrial control systems, smart buildings, monitoring and proactive maintenance

A note on edge computing vs IoT: A common misconception is that edge and IoT are synonymous. Edge computing is a topology—and location-sensitive form of distributed computing, while IoT is a specific use case instantiation of edge computing.

Let's take the example of retail organizations. Over the past decade, retailers have seen dramatic shifts in consumer buying behavior as shoppers move online to research products, read consumer reviews, compare prices, and purchase. Customers demand a great digital experience as well as an outstanding in-store retail experience from all retailers. These experiences need to work seamlessly and in tandem in order to make the omnichannel vision a reality.

The in-store experience is characterized by stateful applications running at the edge (the store being the far edge location).

Applications in retail that would typically run in-store (edge) include:

- ***HR applications such as employee attendance management:*** These are used throughout the day by the various employees that badge in and out of the store. These applications need to be continuously available even if the store goes "offline" and cannot connect to the cloud.
- ***Item lookup (pricing and store locations):*** The product and pricing lookup apps would need to handle a high volume of mostly reads. Price adjustments may be done infrequently (say weekly). The adjustments themselves are computed in the cloud and pushed down to the stores. But, higher frequency of updating inventory and prices may be desirable, and this requirement can change as the apps evolve. Item locations change infrequently (i.e. a few items per week or month).

- ***Real time video for cashierless shopping:*** Video analysis will typically apply an existing machine learning (ML) model to data in the store in real-time. The model is trained using data from several stores by an application running in the cloud, and the trained model is then pushed to stores.

- ***Point of sale systems (POS):*** These are real-time applications that perform end-user initiated transactions. Most data is cached in-store in this case, and there is no need to access anything in the cloud (because this system should be able to run independently, even when the network is partitioned).

- ***Miscellaneous applications:*** Other applications that may be needed include security monitoring on aisles, couponing, and tax computations at the edge.

Retail applications that typically run in the cloud include:

- ***E-commerce:*** The digital shopping experience that includes browsing products, adding to cart, checking out, billing, etc.

- ***Post-purchase experience:*** This includes shipment tracking, notification of location and delivery, handling product returns, etc.

- ***Computing optimal item pricing:*** Item pricing may be computed in the cloud and pushed to both the e-commerce app as well as the stores. This category can include other business intelligence and strategy applications.

- ***Item grouping:*** Finding groups of items that should be sold as a bundle to maximize revenue.

Cloud and edge applications are not isolated from each other. Examples of retail interactions between the cloud and edge:

- Buy online, pick up in store (BOPIS)

- Inventory check of an item in stores nearest the customer

- Apply the latest pricing and discount information from the cloud in stores

Applications running at the edge are typically customer or frontline worker facing. They use data that is generated at the edge location. User-facing applications demand low latency access and continuous availability even if the network goes down, making them great candidates for running at the edge.

Design Considerations for Data in Edge Environments

When designing the architecture for stateful edge applications, there are some key principles and design patterns to keep in mind.

Data Lifecycle

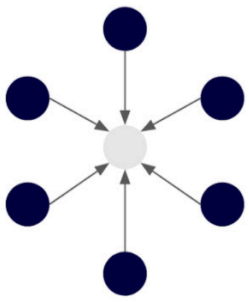
It's critical to take account of where the data is produced, what needs to be done with it, and where it is to be consumed. For example, will it be analyzed, will it be stored and then forwarded, or will it qualify for long-term local storage? Moving data across regions brings high latency and low throughput. In contrast, leaving data close to where it is produced and consumed brings low latency and high throughput.

Data is typically replicated between the cloud and edge locations, or within a tier. Common deployment patterns include:

- ***Hub and spoke pattern***: In this pattern, data is generated and stored in the edges, and the central cluster in the cloud aggregates data from edges. This is a common pattern in IoT use cases and retail, where devices or store locations generate data which is then aggregated in the cloud for analysis.

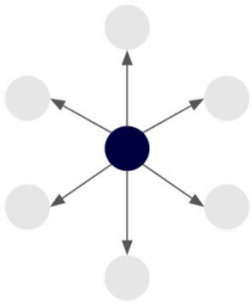
- ***Configuration pattern***: Data is stored in the cloud, with read replicas at one or more edge locations. A good example of this is configuration settings for devices.

- ***Edge-to-edge pattern***: Another common pattern is where data needs to be synchronously or asynchronously replicated, or partitioned within a tier. Examples of applications that use this pattern include vehicles traveling across different edge locations, mobile users in roaming mode, or users traveling between countries and making financial transactions, etc.



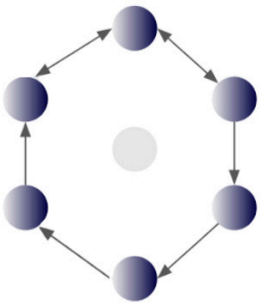
HUB AND SPOKE PATTERN

Replication from edges to the cloud



CONFIGURATION PATTERN

Replication from the cloud to the edges



EDGE–TO–EDGE PATTERN

Replication within a tier

Application Workload Types

Stateful edge applications typically are a combination of the following workload types:

1. ***Streaming data:*** Streaming data is data coming from devices, users, and "things", such as vehicle telemetry, location data, etc. This data may need to get sanitized before it can be used. Streaming data requires high write throughput and fast querying.

2. ***Analytics over streaming data***: Real time analytics applied to streaming data to generate alerts. This should either be supported natively by the database, or by using Spark or Presto in conjunction with the database.

3. ***Event data***: Events computed on raw streams that then get stored back in the database with ACID guarantees.

4. ***Smaller data sets with heavy read-only queries***: This includes configuration and metadata workloads.

5. ***Transactional, relational workloads***: Examples include identity, access control, security and privacy where transactional semantics are important.

6. ***Full-fledged analytics of data***: Some applications need to analyze data in aggregate across different locations.

7. ***Workloads needing long term data retention***: In some cases, organizations want to store data in a warehouse or data lake in order to analyze trends and events over longer time periods, or for audit and compliance purposes.

A combination of the above workloads can run in different locations - the near/far edge or cloud. But certain workloads have affinity to certain locations based on scale. Workloads 1- 4 typically occur on the edge while 5 - 7 occur in the cloud.

Scaling Needs

Another key factor to consider is how fast the data is growing. How many users, devices, etc. will generate data? How much compute power is needed to process the data? For example, edge locations typically do not have the compute and storage resources to run deep analytics on vast amounts of data. Also, OLTP databases at the edge may need to scale throughput to handle massive write volumes from devices.

Latency and Throughput Needs

How much data will be written or read? Will the data come in bursts or as individual data points? How quickly does it need to be available to users and applications? For example, with real time applications such as connected vehicles and credit card fraud detection, it is not feasible to send telemetry or transaction data back to a cloud application to determine a course of action. In these cases, real time analytics is applied to raw data in edge locations to generate alerts.

Network Partitions

Poor network connectivity is a reality for many near and far edge locations. Applications should take into account how to deal with network partitions. Depending on network quality between the edge and the cloud, different operating modes can occur:

- ***Mostly connected:*** The applications can connect to a remote location to perform an API call (i.e. to lookup data most of the time. A small fraction of these API calls may fail because of network partitions (i.e. a few seconds of partitions over several hour timespan.

- ***Semi-connected:*** In this scenario, there could be an extended network partition lasting several hours. Applications would need to be able to identify changes that occurred during the partition window, and synchronize their state with the remote applications once the partition heals.

- ***Disconnected:*** The predominant operating pattern in this case is that the applications run independently of any external site. There may be an occasional connectivity, but that is considered the exception rather than the norm.

Applications and databases running in the far edge should be designed for disconnected or semi-connected operation. Near edge applications should assume semi-connected or mostly connected operation. The cloud operates in the mostly connected mode. That said, when a public cloud service experiences an outage, the impact is severe and can last many hours.

Other Failures

In addition to network partitions, infrastructure outages can be quite common in various locations as well. At the far edge, everything from node/pod failures to a complete regional outage can be common. At the near edge and in the cloud, while node or pod outages are common, applications can use racks and zones for higher resilience. But, even with this fault isolation in place, region-level outages can occur.

Not all failures need to be outages. Other types of failures include resource contention and resource exhaustion.

Software Stack Considerations

It is important to think about the agility and ease of use when picking components for the software stack. Business services involve a suite of applications, so engineering teams need to design for rapid iteration on applications. One way to achieve this is to use well-known frameworks that enable instant developer productivity (Spring, GraphQL, etc), and a well-known and feature-rich database that developers already know well and is open source. Open source software enables deployment flexibility (e.g., PostgreSQL, YugabyteDB).

Security

Regardless of whether an application is running in the cloud or at the edge, security is paramount, especially since there is a large surface area of attack given the inherently distributed nature of the architecture. It is important to think about least privilege everywhere, zero trust, zero touch provisioning for all services and components.

Some of the other specific security aspects that come up are listed below:

- Encryption in transit
- Encryption at rest
- Multi-tenancy support at database layer and per-tenant encryption
- Regional locality of data to ensure compliance and thinking about any geographic access controls that go with it

Picking Data Platforms

The cloud and edge locations have different requirements for data platforms. These requirements are driven by the workloads that typically run in these tiers.

OLTP vs OLAP

In near and far edge locations, OLTP databases are more important than OLAP, because these locations deal with streaming data that needs to be stored for fast querying, building indexes and applying constraints. They need to update erroneous data, handle out of order data arrival, backfills, etc. Limited compute power at the edge restricts the level of analytics that can be run on OLAP databases. Organizations can choose to run Spark or Presto on OLTP databases in scenarios where they need analytics at the edge. Cloud environments typically need both OLTP and OLAP databases.

OLTP Requirements at the Edge vs the Cloud

When it comes to OLTP databases, workload requirements and resource constraints dictate database needs at the edge and in the cloud.

Near and Far Edge

- ***Small footprint:*** Given the limited resources available at edge sites, organizations need to be able to deploy the database in small footprints
- ***Hybrid/private cloud:*** The databases need to be able to run on hybrid/private cloud infrastructure, since cloud services are typically not present at the edge. In some cases, tier 2 clouds, CDNs and public cloud services that specialize in verticals may be available at the edge.
- ***Operational simplicity:*** The database should be easy to deploy and maintain with limited onsite resources.
- ***Disconnected mode:*** The database should be able to operate in the event of a network partition.

- ***Operational simplicity:*** The database should be easy to deploy and maintain with limited onsite resources.
- ***Disconnected mode:*** The database should be able to operate in the event of a network partition.
- ***Kubernetes ready:*** The ability to run in Kubernetes environments is a plus because it simplifies operations and allows organizations to keep things consistent across different clouds. If this can be achieved through some other means, then it may not be an issue.
- ***ACID transactions and relational data modeling:*** Event data workloads need transactional databases because event data cannot be lost. ACID guarantees are important. Relational data modeling is also preferred for building applications.
- ***Scale and performance:*** Telemetry data needs scale and performance – traditionally this is why engineers try to use NoSQL databases. But, if a relational database can scale and deliver high performance, then it is an ideal choice.
- ***Time to live (TTL):*** It is sometimes useful to have the database retire older data in order to deal with rapid data growth.
- ***Bi-directional data transfer to/from the cloud:*** Edge applications are typically not isolated from the cloud, so having the ability to transfer data to and from the cloud efficiently is important.

Cloud

- ***Multi-cloud:*** Organizations should be able to run the same database in one or more public cloud services of their choice without restrictions. Committing to a single cloud is risky, especially since different cloud services are not at the same level of maturity when it comes to features and geographic availability. For example, applications may be geo-distributed and may need to comply with local data residency requirements. What happens if the chosen cloud provider does not have a regional presence in a particular country?
- If private data centers are available to host cloud applications, then support for hybrid deployments (on prem and public cloud) is important as well.
- ***Operational simplicity:*** A fully managed multi-cloud DBaaS can eliminate the operational burden of deploying and maintaining a database management system in the cloud.
- ***Integrations:*** Applications running the cloud need to be able to send data to, or receive data from, other databases and data warehouses.
- ***Scale:*** Databases in the cloud need to be able to scale out/up and scale in/down, on demand, without downtime or operational overhead, using available cloud resources.

Distributed SQL for Edge Applications

A distributed SQL database offers a viable platform for running transactional applications in the cloud and at the edge. You can run the same database in different tiers, making it operationally simpler. Let's look at some of the characteristics of a distributed SQL database that make it a great fit for edge environments.

Continuous Availability

A well architected distributed SQL database is designed for resiliency. There are no single points of failure, and data is automatically replicated across nodes of a cluster. Critical services remain available during node, zone, region, and data center failures with fast failover, as well as during system maintenance.

Horizontal Scaling

The database cluster can be scaled out and in with zero impact simply by adding nodes to it. This allows enterprises to scale a distributed SQL database on demand to meet a growing need for capacity, throughput, or number of database connections.

Flexible Geo-Distribution

A distributed SQL database offers powerful synchronous and asynchronous replication within a region, and between the core and edge. The database also has built-in geo-partitioning and data pinning capabilities for compliance, as well as read replicas for hub-spoke topologies.

Advanced RDBMS Features

A distributed SQL database delivers familiar relational data modeling that allows developers to build data-driven applications. Well-designed distributed SQL databases offer a comprehensive set of advanced RDBMS features that developers may be familiar with.

YugabyteDB: Best-in-class Distributed SQL for the Edge

YugabyteDB is a cloud native distributed SQL database for transactional applications. The database is 100% open source and built to address the needs of cloud and edge applications.

Deployment Flexibility

YugabyteDB runs in public, private, and hybrid cloud environments, on VMs, containers or bare metal. Organizations can deploy the database in any Kubernetes environment. It is also available as a multi-cloud fully managed DBaaS for a frictionless experience. YugabyteDB offers the most wide range of replication and geo-distribution options among distributed SQL databases.

High Performance

YugabyteDB can handle high throughput, low latency I/O at the edge. It is proven in production to scale beyond 1 million transactions per second and thousands of concurrent connections.

Operational Simplicity

Organizations can use the self-managed or fully managed DBaaS offerings of YugabyteDB to simplify operations at the edge and in the cloud. The database also integrates with other data sources or sinks, allowing data engineers to build pipelines for machine learning, analytics, long term storage, and disaster recovery.

PostgreSQL Compatibility

YugabyteDB is not just wire compatible with PostgreSQL, it is code compatible. The database also offers a comprehensive set of advanced RDBMS features including triggers, functions, stored procedures, and strong secondary indexes. This allows developers to be immediately productive with the familiar interface and the rich ecosystem of PostgreSQL compatible frameworks, applications, drivers, and tools.

Security

YugabyteDB is built from the ground up with data security in mind, enabling organizations to maintain a robust security posture even with a more distributed footprint. YugabyteDB offers features such as data encryption at rest and in flight, multi-tenancy support at the database layer and per-tenant encryption, and regional locality of data to ensure compliance as well as manage geographic access controls.

Conclusion

The rise of edge computing is a paradigm shift in the way applications are built and deployed to cater to the needs of an increasingly decentralized world. There is no one-size-fits-all database reference architecture that works for all applications in this environment. Depending on the requirements of the application and tradeoffs involved, enterprises will choose different topologies to meet their needs, and change the topologies when needs change.

Distributed SQL databases offer a powerful and versatile data layer for running applications in both the cloud and edge environments.

yugabyteDB

Get in Touch

www.yugabyte.com | contact@yugabyte.com

yugabyte.com/slack

twitter.com/yugabyte

instagram.com/yugabyte

linkedin.com/company/yugabyte