Cockroach Labs

CockroachCloud Evaluation Guide

This 4-step guide provides an overview of CockroachDB-as-a-service — CockroachCloud — and a basic approach for evaluation of the database. It is intended to help you to focus your time on the criteria that's important for your explicit application(s).

This guide will help you:

1. Discover if CockroachCloud is right for you,
2. Learn CockroachCloud,
3. Establish your success criteria, and
4. Run your evaluation.

1. Discover CockroachCloud

Is your application right for CockroachCloud? If you have an OLTP (Online Transaction Processing) workload, CockroachCloud may be right for your application. It eases scale and delivers a highly available database built for relational transactional workloads. It is a good choice for small single region applications but can also grow with you to be deployed in multiple regions without sacrificing latency or consistency. And if your application has deep requirements for strong data consistency, and/or data locality, then CockroachCloud might be your best choice.

What is CockroachCloud?

CockroachCloud delivers CockroachDB as a service. It extends the benefits of CockroachDB with self service and managed service offerings to make your deployment fast, smooth and simple. CockroachDB is a cloud database, architected and built from the ground up to deliver a single or multi-region, active-active, strongly consistent SQL Database that gives developers:

- **Simple, Consistent SQL**

  Use the tools and the language you use today and eliminate data anomalies
- **Elastic, Efficient Scale**

  Eliminate need for complex database operations and scale whenever and wherever you want
- **Bulletproof Resilience**

  Ensure uninterrupted, high performance queries during failure of a server, rack or region
- **Multi-region, Multi-cloud**

  Start small and grow to meet your users no matter where they are with a single database
- **Distributed & Cloud-native**

  Align your containerized apps and services with a database architected for modern distributed systems

CockroachCloud is a fully managed service where hardware, deployment, and ongoing management of your database are covered by our SRE (Site-Reliability Engineering) team, so that you can focus on building your application and not the operational complexity of the database. Here is what makes CockroachCloud unique compared to other database-as-a-service offerings:

User Experience	Managed Service	Self-Service
Developer-first features	Capabilities managed by our SREs	Features available in the console
SQL interface Utilize well-known Postgres drivers and ORMs for your application Strong consistency (serializable isolation) Secondary indexes JSONB support Data geo-partitioning Geospatial queries Change data capture	Always on database Multi-region writes Encrypted storage, connections and backups Health monitoring Network load balancing SOC II	Cluster creation VPC peering & IP allow listing monitoring Add/remove nodes & users User authorization Restore from backups SQL client

Why use CockroachCloud?

CockroachCloud can be used for everything from a simple log of actions to the most complex financial transaction processing applications. It is built to deliver an experience that will grow with your application and delivers familiar SQL that is wire compatible with PostgreSQL. You might consider CockroachCloud if your company has any of the following common goals:

- Build new applications
- Modernize Data-Intensive Applications
- Move to the Cloud
- Build a Global Application Architecture
- Ensure Data Privacy & Comply with Regulations

How is CockroachCloud used?

CockroachCloud is great for Online Transaction Processing (OLTP) applications. These are systems of record type workloads that run your business, such as authenticating users, handling orders, transacting payments, and more. New use cases are developing daily because of innovations with CockroachDB to work in highly distributed, geographically dispersed environments.

Common use cases

- **General-Purpose OLTP Database** that serves as the primary data store for OLTP apps and services
- **System of Record** for managing financial transactions, inventory, order management, payments, etc
- **Identity & Access Management** for storing and serving authentication/authorization service data such as user credentials, access tokens, license keys, etc.
- **Locality Aware Applications** for GDPR compliance, eCommerce, customer service, chats, etc.
- **Metadata Store** for holding application metadata and/or workflow information
- **Key Value Store** for storing and accessing data through key value lookups
- **Spatial Database** for storing and querying location, geography, and geometry-based data

Financial Services	Retail & eCommerce	Media & Entertainment
• Payments • Card authorization • Digital banking services Sample customer: Nubank	• Gift cards • Inventory • Shopping carts • Order management Sample customer: LUSH	• Financial ledger • Betting records • Customer loyalty • Customer relationship management Sample customer: Kindred
Gaming	**Manufacturing & Logistics**	**Startups**
• Player chat services • Identity & access management • Payments Sample customer: Insights	• Supply chain • Inventory • Distribution and delivery Sample customer: Shipt	• Identity & access management • Application metadata • IT asset tracking Sample customer: MyMahi

Less ideal use cases

In time, the following use cases may work with CockroachCloud, but as of today they are not a great fit:

- OLAP (Online Analytical Processing) & Data Warehousing
- Advanced Time Series
- Full Text Search

2. Learn CockroachCloud

If you have an application in mind for CockroachCloud, then the next thing to do is learn more about CockroachDB. We offer free training and several handy tutorials to ramp you up quickly. A great place to start is free courses on Cockroach University which can walk you through the aspects and features of the database. Additionally, we have a number of tutorials to help ramp you up a variety of topics.

Training on Cockroach University

- Getting Started
- Python Developers
- Java Developers (coming soon)

Tutorials

- QuickStart (30 day Free Trial) or Create a Cluster
- Deploy a Flask App in CockroachCloud
- Stream Data to Snowflake
- Serializable Transactions
- JSON Support
- Geospatial

Sample code

This Django Tutorial shows how to build a simple application with CockroachDB. These Hello World Apps are tutorials that work specifically for CockroachDB but can be adapted to work with CockroachCloud. Instead of using `cockroach demo`, utilize the CockroachCloud command line interface (CLI) instead. These tutorials include: Java, Go, Python, Ruby, Node, C/C++, C#, Clojure, Rust, PHP, TypeScript

CockroachDB internals

CockroachDB turns all database objects such as tables, indexes, etc. into ranges. You can think of a range as a shard of data. A range will have replicas that are part of a range set. The number of replicas in a range set is defined by the replication factor and it is typically an odd number. Ranges have a size constraint that allows ranges to grow, split, merge and be moved around the cluster. Ranges can further be controlled by creating a zone which tells these ranges where to be placed, what the replication factor should be and a few other settings. Each range set will also have a leader which is called the leaseholder for reads, and the RAFT leader for writes.

All reads, go through the leaseholder and all writes go through the RAFT leader. Typically both the leaseholder and RAFT leader are on the same range in a range set.

Multi-region planning

Multi-Region deployments take a fair amount of design and testing to achieve your desired performance and resilience needs. Topology patterns outlined in the CockroachDB documentations are helpful in thinking about how to lay out the schema and database objects in your database.

In a nutshell, database reads are generally easy to make fast in CockroachDB. Writes are bound by the distance between where range replicas are. And Lastly, resilience depends on your survivability tolerance when a node, availability zone or region goes down. The tug of war with multi-region patterns is performance vs resiliency which is important to think about as you design your system.

Default setup

When you create a multi-region cluster in CockroachCloud, here is the default arrangement you are given.

- Each region for your cluster will have a network load balancer
- The default replication factor for all user objects (tables & indexes) is 3
- Follow the Workload topology is the default setup, which means if you create a table in your local region, the table may not be localized to your region.

Topology considerations

It's possible the default setup may not be useful for your application. Fortunately there are different architectural schema topologies that can be utilized for defining what's more important to you regarding performance vs resilience. Topologies are typically managed by changing the DDL definitions for your database objects. Below is an outline of the options you have for each topology.

	Reads	Writes	Resilience
Follow-the-workload *(Default in CockroachCloud)*	Fast in leaseholder region, slower elsewhere	Modest in RAFT leader region, slower elsewhere	Node, AZ or Region
Geo-partitioned leaseholders	Fast in primary region, slower elsewhere	Fast in primary region, slower elsewhere	Node, AZ or Region
Geo-partitioned replicas	Fast in primary region, slower elsewhere	Fast in primary region, slower elsewhere	Node, AZ only

Duplicate indexes	Fast in all regions	Slow	Node, AZ or Region
Follower reads	Fast in all regions, but 4.8 seconds state	Modest in RAFT leader region, slower elsewhere	Node, AZ or Region

Replication factors

Replication factors are controlled at the range level for a database, table or row. Replication factors are not at the cluster nor node level. The replication factor does not have to match the number of nodes either. Therefore, different groups of objects in your cluster can have different replication factors. For example, let's say we have a 5 region cluster used for Identity Management. Select queries to the USER table may need to be fast so we can utilize the Follower Read pattern, create a replication factor of 5 for the USER table so that a replica exists in each region for fast reads.

Replication factors can increase your overall resiliency at the expense of having slower writes. Additionally, replication factors for multi-region clusters can assist with having local replicas available for speedy Follower Reads patterns as well.

Application adjustments

Without changing any application code moving a single region application to a multi-region setup, you can expect performance to be slow until you make some application and database adjustments. When thinking about multi-region, think about where data will be accessed and transacted locally. If you can pin leaseholders in the regions where data needs to be accessed and transacted, performance will be considerably improved.

Considerations

- Each region can have it's CockroachCloud VPC be peered to your application network
- Applications generally should run in each region where CockroachDB is located and communicate directly to the supplied connection URL in the CockroachCloud for that local region. The connection URL points to a regional network load balancer for the cluster.
- Global Load Balancers can be considered for routing users and requests to the nearest region. CockroachCloud does not provide a Global Load Balancer service.
- If running a workload or benchmark, a proper setup would be to run the workload or benchmark with a separate client(s) in each region. Additionally, arranging the workload to work with regionalized data that may have been setup using the Geopartioned Leaseholder topology will increase the read and write performance of the workload. This requires configuring both the app and database to be arranged accordingly.

Migration planning

Migrations do take more effort in an evaluation as opposed to building a brand new application. If planning a migration, we have a number of documents that can guide you. The dialect for CockroachCloud is very similar to Postgres, so Postgres migrations are fairly smooth. Below are links to documents that can help you plan a database migration.

- Migrate from Postgres
- Migrate from Oracle
- Migrate from MySQL

Limitations

Like any technology, CockroachDB does have some known limitations that should be considered before starting your evaluation.

General limitations	See this link for general CockroachDB limitations.
Unsupported Postgres features	User Defined Functions, Triggers and Stored Procedures are currently not supported. Learn more.
Large database objects	We recommend object sizes to be less than 1MB. A better pattern is to leave those objects in another system (i.e. object storage) and have a CockroachDB maintain a reference or pointer to that object.
Change data capture	using Kafka client authentication for TLS or Kerberos is not supported today.
Adding regions to existing cluster	Nodes can be added and removed but regions cannot be added or removed.

3. Establish your success criteria

Proper success criteria dictates how your evaluation should be planned. Additionally, making success criteria measurable drives your evaluation towards a common goal. Most importantly, success criteria should pertain to multiple constituents that have to interact with the CockroachDB database. Typically this includes, and is not limited to, developers, administors, operators, architects and security administrators

Key areas to evaluate

	Criteria	Docs	Metrics & Checks
Cluster operations	Scaling In/Out	Add/Remove Nodes	[] Add a Node [] Remove a Node
	Backup & Restore	Restore from backup	___ Restore Time [] Restore Success
	Server Maintenance	Rolling upgrades	[] Successful Upgrade
SQL	Schema changes	Online schema changes	__ Query Execution Time
	Data consistency	Serializable isolation	[] Successful Test
	Query latency	Best Practices, Explain Plans & SQL Diagnostics	__ Query Execution Times
	Throughput	Benchmarks & Performance Overview	__ Transaction Per Second
Security	Secure your cluster	Authentication, Authorization & Audit	[] Passes Security Tests
	Encryption	Enabled by default	☑
Integrations	App Connectivity	Deploy a sample app	[] App Connects
	Network Auth	VPC Peering and IP Allow Listing	[] CockroachDB is reachable
	Migration	Migration Overview	[] Migration Successful
	Data Integration	Snowflake Integration	[] Integration completed

Get your environment

Selecting nodes and regions

Single Region environments can have 1 (development) or 3+ (production) more nodes in a region you specify. Reads and Writes will perform well and data will evenly distribute amongst the nodes.

Multi Region environments can have 3 or more regions. Read and writes will depend on the topologies you pick as well as the distance between the regions. Careful planning is needed to attain the performance and resiliency that is desired. If you would like a Multi-Region environment, create a service request here.

Scaling Considerations: Horizontal Scale can add more storage, compute, throughput and higher resilience. Vertical scaling can add more performance.

Cluster sizing

Here is a simple back-of-the-napkin calculation you can do to make an educated guess on the number of nodes you may need. This calculates the number of nodes you may need based purely on storage and without any compute assumptions.

- Raw data uncompressed in GB
- * Replication factor (Default is 3)
- / Disk Size Per Node (See “Hardware Per Node” when you Create a cluster)
- **= Number of Nodes**
- **Example: 250GB Raw Data** * Replication Factor of 3 / 150GB Disk Drive = 5 nodes

Data Compression and Head Room typically net each other out in the storage sizing calculation.
Deciding on the amount of vCPUs nodes needed to support your workload can be calculated by thinking about the amount of active connections, not idle connections, your workload needs to sustain. For each vCPU in a cluster, 4 active connections can generally be sustained. Consequently Cockroach can have a lot of idle connections open.

You can always start with a smaller cluster and add more if you need more compute and storage.

4. Run your evaluation

Create your CockroachCloud environment

Register for an account. You can follow the directions for a QuickStart (30 day trial) or Create a Dedicated Cluster.

Set up your application with CockroachCloud

Connect to your cluster

Instructions are here for connecting your application to the CockroachCloud service. In order for your application to connect, you can use either IP Allow Listing or VPC Peering to connect your applications VPC to the CockroachCloud VPC. The connection string provided is the network load balancer that will connect you to the nodes in the CockroachDB cluster.

Create your test application

If you are creating your test application from the ground up, be sure to leverage our code samples here. There are best practices in the sample code for managing your interaction with the driver, handling errors and writing code.

Migrate your application

If you're migrating an application from another database, you will most likely run into some small SQL dialect issues that you need to remediate. The CockroachCloud migration docs are very helpful for understanding SQL syntax and the steps for migrating your schema and data.

Transaction retries

One key design consideration is handling transaction retries. Because CockroachDB uses Serializable Isolation, contention can be created if SQL, schema, data and application code are unoptimized. This can cause transactions to retry and be sent back to the client application for handling. Applications should include transaction retry logic. More details on writing this logic can be found here and error code references can be found here.

Gather your evaluation results

Again, you can use the Success Criteria and record your results in the last column. Hopefully you've achieved your goals and are ready to move forward with CockroachDB. If your evaluation did not go as planned or you need technical assistance, see the next section.

Technical support

During an evaluation, you may hit a bug, a performance issue or a design challenge. There are several resources you can utilize to help you get an answer.

Ask questions in the Cockroach Labs Slack Community. The #CockroachCloud channel is a great place to ask a question. There are also announcements, blogs, and do feel free to leave us product feedback. Several Cockroach Labs employees frequently monitor the Slack community.

In addition, you can post questions in our Forums which are also monitored by Cockroach Labs employees.

Issues and bugs can be logged in our Support Portal. Service level agreements are in place to respond on the severity level for customers and non-customers.

Finally, if you want to speak to someone directly, you can always contact your Account Team to get more assistance with your efforts.