CockroachDB

Distributed SQL at scale for the cloud native future
A Journey to the (global) cloud

All organizations (of all sizes) are on a journey to become cloud native with global scale, but are at various stages:

- **MONOLITHIC, SINGLE DC**: Startup or move to cloud
- **DISTRIBUTED, SINGLE DC**: Scaling in cloud
- **DISTRIBUTED, MULTI-DC**
- **GLOBALLY DISTRIBUTED CLOUD**: Business Continuity

Cockroach Labs enables organizations to reliably transition their most valuable and vital transactional data & workloads to the cloud.
Another Database?

<table>
<thead>
<tr>
<th></th>
<th>RELATIONAL</th>
<th>NOSQL</th>
<th>DISTRIBUTEDSQL</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Single instance, transactions on legacy infrastructure</td>
<td>Global, optimized for read access to data</td>
<td>Architected for transactional cloud applications</td>
</tr>
<tr>
<td>Scale</td>
<td>Difficult manual shard or asynchronous replication</td>
<td>Automated for read only access of data</td>
<td>Simple, global scale for reads and writes</td>
</tr>
<tr>
<td>Resilience</td>
<td>Active passive failover creates RPO lag</td>
<td>Distributed data allows for quick global reads</td>
<td>All active redundancy eliminates RPO</td>
</tr>
<tr>
<td>Transactions</td>
<td>Ensures consistent transactions</td>
<td>Limited transactional capability</td>
<td>Serializable isolation ensures consistency</td>
</tr>
<tr>
<td>Cloud</td>
<td>Architected for legacy infrastructure</td>
<td>Architected for web, read only infrastructure</td>
<td>Architected for cloud-native apps</td>
</tr>
</tbody>
</table>

The age of cloud scale and advent of microservices requires a new approach for the relational, transactional database
CockroachDB delivers a **Distributed SQL database**

An **EVOLUTION** of the relational database for cloud native, distributed transactions

1. Implements standard SQL interface
2. Eases operational complexity of scale
3. Geo-replicated, always on and resilient
4. ACID compliant distributed transactions
5. Ties data to a location
CockroachDB: a unique distributed architecture

**self contained, aware nodes** participate in global cluster

Each node within a cluster is self-contained and has locational awareness of itself and others.

Every node is a CONSISTENT gateway to the entire database

- Intelligence packed with data
  - Management & optimization
  - Standard SQL engine
  - Enterprise security
  - Ecosystem integration
CockroachDB: a unique distributed architecture

global database cluster coordination and logic

Spin up a node anywhere (public and private clouds) and then point it at the cluster, which takes care of:

- Coordination & consensus for queries/transactions
- Replication, repair & rebalancing of data across cluster upon addition/removal of nodes
- Attach location to any data to set domiciling & replication constraints

Inherently multi-cloud
CockroachDB: **Scale** your data not your complexity

Replication, repair & rebalancing of data across cluster upon addition or removal of nodes

1. To expand capacity, simply add new nodes to the cluster & data is automatically rebalanced
2. Automated balancing eliminates need for manual sharding and complex resharding
3. Balancing optimizes server efficiency (storage and compute)

Cluster automates balance of data evenly across all nodes
CockroachDB: Always on and naturally resilient

Your data is always on and always available

1. On failure, data is efficiently redistributed and replicated across nodes within clusters
2. CockroachDB eliminates the need for costly active/passive or complex CDC architectures needed for redundancy
3. Minimize impact & recovery time from failure, with Cockroach RPO is zero

...AND rolling upgrades!
CockroachDB: Global **consistency**, immediate not eventual

Ensures consistency across distributed transactions

**CockroachDB** uses clocks and concurrency controls to deliver **full ACID** transactions at scale even in a distributed environment

Serializable isolation protects from write skew and dirty reads

*CockroachDB: Serializable isolation in a distributed SQL database*

*transactions may not physically execute serialized in time, rather they execute as if they have. They are guaranteed to appear serialized*
CockroachDB: Tie your data to a location

Geo-partition your data to set domiciling & replication constraints

1. Tie explicit “ranges” of data to a geography or any address at the table or row level
2. Comply with privacy regulation OR have data follow a user to reduce latencies
3. Tie data to explicit clouds and maintain global access to all nodes throughout cluster

Data can be tied to any place (country, cloud, etc.)
CockroachDB: Inherently multi-cloud

Implement a globally consistent database across clouds and even on premise
CockroachDB: your bridge to the cloud

CockroachDB provides consistency, resiliency and locality at scale meet the needs of heavy read/write distributed transactional workloads

Modernization

• Mainframe replacement
• Migration of databases to cloud/distributed
• Migration of application to microservices
• Consolidation/simplification project (ETL and multi dbs/systems reduction)

Net New & Cloud Applications

• New application: system of record
• New application: metadata layer
• Geo-partitioning for low latency access
• Regulatory compliance

Enables a future data architecture and your cloud-native future
CockroachDB and Kubernetes

Common distributed architecture

- Natural fit for pods and orchestration
- Helm chart available eases deployment
- Multi-region and globalscale
- Geotagging within CRDB helps tie compute to data and locality

CockroachDB uses the Storage class and PV claim to mount a volume within a cluster and then builds on stateful sets, so we naturally inherit the controls and power of Kubernetes
CockroachDB
Cloud native distributed SQL for the cloud native future

Cloud Neutral
Build across on-prem, cloud, hybrid cloud and multi cloud environments

Open Source
UN-opinionated and community driven so you are not tied to any cloud

DOWNLOAD NOW!
Cockroach Labs

Product: CockroachDB, a distributed SQL database that survives outages and eases scale of cloud applications

Founded: 2015 by ex-google engineers: Spencer Kimball, Peter Mattis & Ben Darnell

Investors: include Benchmark Capital, Index Ventures, Redpoint and GV

Offices: New York City (headquarters), San Francisco & Seattle