CockroachDB The definitive guide – proposed outline

[Part 1: Introduction to CockroachDB](#_Toc64373665)

[Chapter 1: Why CockroachDB? – 25 pages](#_Toc64373666)

[Chapter 2: CockroachDB Architecture – 30 pages](#_Toc64373667)

[Chapter 3: Getting started – 20 pages](#_Toc64373668)

[Chapter 4: CockroachDB SQL – 30 pages](#_Toc64373669)

[Part 2: Developing applications](#_Toc64373670)

[Chapter 5: Schema design – 40 pages](#_Toc64373671)

[Chapter 6: Application design and implementation – 25 pages](#_Toc64373672)

[Chapter 7: Integrating with other systems - 20 pages](#_Toc64373673)

[Chapter 8: Query Tuning – 40 pages](#_Toc64373674)

[Part 3: Administration and deployment](#_Toc64373675)

[Chapter 9: Planning a deployment -35 pages](#_Toc64373676)

[Chapter 10: Production Deployment - 40 pages](#_Toc64373677)

[Chapter 11: High availability and disaster recovery – 25 pages](#_Toc64373678)

[Chapter 12: Security – 20 pages](#_Toc64373679)

[Chapter 13: Administration – 30 pages](#_Toc64373680)

[Chapter 14: Cluster and Node optimization – 40 pages](#_Toc64373681)

# Part 1: Introduction to CockroachDB

## Chapter 1: Why CockroachDB? – 25 pages

In this chapter, we briefly review the history of database technology from pre-relational through relational and to “NoSQL”. We’ll see how the emergence of Web 2.0 challenged the RDBMS model, leading to “NoSQL” distributed alternatives, and how a new breed of distributed SQL systems emerged in response to the limitations of these systems.

### A brief history of databases

### Distributed database systems

### The CockroachDB approach

### CockroachDB benefits

### CockroachDB use cases and notable deployments.

## Chapter 2: CockroachDB Architecture – 30 pages

This chapter will outline the CockroachDB architecture from APIs and transactional models through to distributed clusters. We’ll review the “familiar” bits: wire protocol, SQL language, transactions, then on to the architecture of a single node server. Finally, we’ll dive deep into the distributed database architecture covering topics such as Ranges, replication strategies, leaseholders, clock synchronization and the role of the Raft protocol.

### The cockroachDB API

### The SQL Engine

### The KV Engine

### Ranges, Nodes and Clusters

### The RAFT protocol

## Chapter 3: Getting started – 20 pages

This chapter will provide a quick start guide for CockroachDB. We’ll show how to startup a single node cluster, how to connect to Cockroach Cloud and how to build a multi-node cluster. We’ll then see how to connect with SQL clients and programming languages and how to use the admin console.

### Creating a local cluster

### Using the CockroachDB cloud

### A multi-node cluster

### CockroachDB clients and drivers

### The admin console

## Chapter 4: CockroachDB SQL – 30 pages

In this chapter, we’ll describe the CockroachDB SQL language at a high level, reviewing the basics of data query, manipulation and definition. We’ll pay particular attention to CockroachDB specific language features.

### The SELECT statement

### Data Manipulation Language

### Data Definition Language

### Functions

### JSON support

# Part 2: Developing applications

## Chapter 5: Schema design – 40 pages

A sound data model is the foundation of a performant and maintainable application. In this chapter, we’ll review the fundamentals of relational schema design, with a particular focus on aspects of schema design that bear on distributed database operations and on advanced CockroachDB features such as column families and JSONB support

### The Fundamentals: third normal form, denormalization

### JSON Support

### Distributed schema design considerations

### Column Families

### CockroachDB indexing

### Views and Materialized views

### Stored procedures (are these in scope for an upcoming release?)

## Chapter 6: Application design and implementation – 25 pages

Like all databases, CockroachDB responds to requests from application code. How an application requests and uses data has a huge bearing on application performance and scalability. In this chapter, we’ll review how an application should work with CockroachDB – including best practices for coding CockroachDB requests and transactional models

### Postgres drivers and ORMs

### Coding best practices: parameterized queries, client-side caching, Array processing, row mode (node) , connection pools

### Transactions and locking patterns

## Chapter 7: Integrating with other systems - 20 pages

Applications don’t usually exist in isolation. Data often flows between systems and databases. In this chapter, we’ll discuss how to load data into CockroachDB and how to integrate CockroachDB with external downstream systems

### Data load facilities

### Change Data Capture

### Integration with Kafka

## Chapter 8: Query Tuning – 40 pages

SQL is a non-procedural language – a SQL statement says what data is to be processed, not how the data is to be processed. Understanding how CockroachDB optimizes SQL statement and how to improve the performance of a poorly performing SQL is a core competency for any developer. In this chapter, we’ll see how CockroachDB optimizes SQL, and how you can view and improve SQL execution.

### Optimizing the optimizer

### Finding slow queries

### Working with explain plans

### Table statistics

### Table scan or index?

### Join optimization

### Sorting and group optimization

### Vectorized execution

### Spatial data

# Part 3: Administration and deployment

## Chapter 9: Planning a deployment -35 pages

The distributed nature of CockroachDB allows for a very large range of deployment topologies. Choosing the right topology requires an understanding of your application’s requirements and the cost and performance implications of various CockroachDB options. In this chapter, we review the steps in planning a deployment and provide an overview of the most common deployment patterns.

### Know your requirements: Latency vs availability vs TCO

### Cloud vs container vs bare metal

### Capacity planning

### Node configuration

### Topology patterns: Follower reads, follow the workload, geo-partitioning, multi-region, Follower reads, follow the workload

## Chapter 10: Production Deployment - 40 pages

In this chapter, we show how to deploy a production CockroachDB cluster. We provide general guidelines on various deployment options with a deep dive into deploying with Kubernetes.

### Deploying CockroachDB with Kubernetes

### Other deployment options: docker swarm, manual

### Managing clock skew

### Cloud specific considerations

### Configuring nodes

### Configuring zones

### Multi-region setup

### Production checklist

## Chapter 11: High availability and disaster recovery – 25 pages

High availability is the only option for modern applications. In this chapter, we’ll see how to configure CockroachDB for a zero-downtime deployment. We’ll also look at contingencies for disasters and how to recover from human errors that damage data.

### Configuring for high availability

### Configuring for disaster recovery

### Human error scenarios

## Chapter 12: Security – 20 pages

In this chapter, we’ll look at CockroachDB facilities for identification, authorization and encryption.

### Layers of security for a CockroachDB application

### Roles and privileges

### Encryption

### SSL

### Single sign-on

### Auditing

## Chapter 13: Administration – 30 pages

If you have created your own deployment of CockroachDB, then there are administrative procedures that ensure that your deployment is stable and performant. Most of these procedures are managed for you in a cloud deployment.

### Monitoring

### Software updates

### Modifying cluster topology

### Troubleshooting

### Managing a CockroachDB cloud deployment

## Chapter 14: Cluster and Node optimization – 40 pages

A complex distributed system creates an additional layer of performance optimization compared with traditional database systems. In this chapter, we’ll look at how to assess the performance of a running cluster and how to relieve bottlenecks and address scaling issues.

### Assessing performance

### Application tuning revisited

### Tuning nodes

### Addressing Cluster bottlenecks

### Scaling out