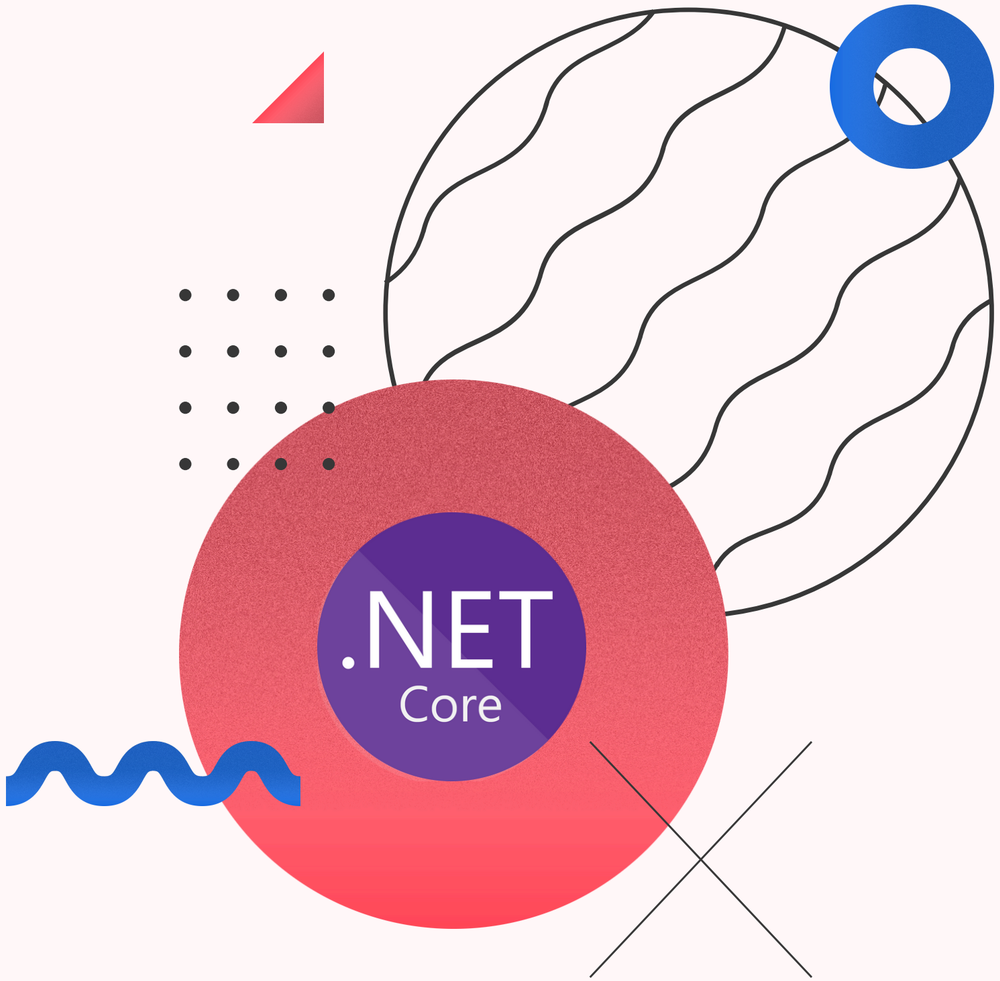
STARTUP HAKK

SQLite

VS

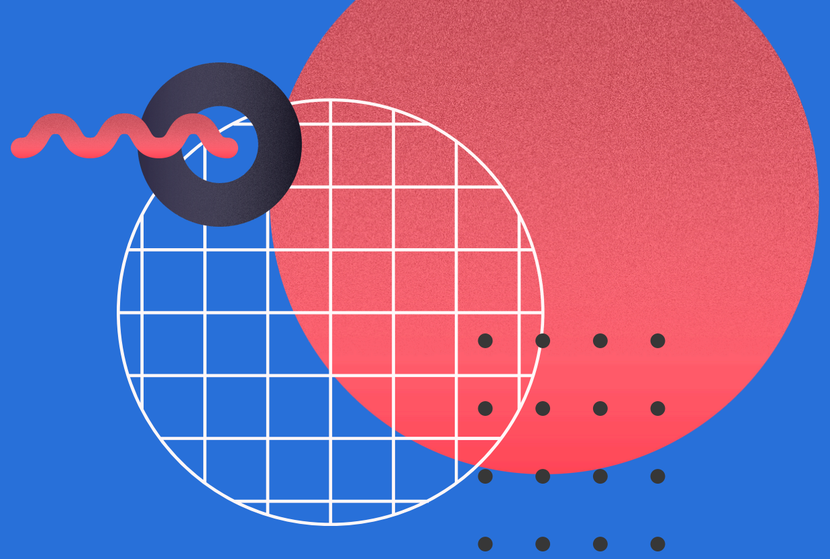
Couchbase



Spencer Thomason

Twitter: @StartupHakk
Facebook: @StartupHakk

SQLite: Pros

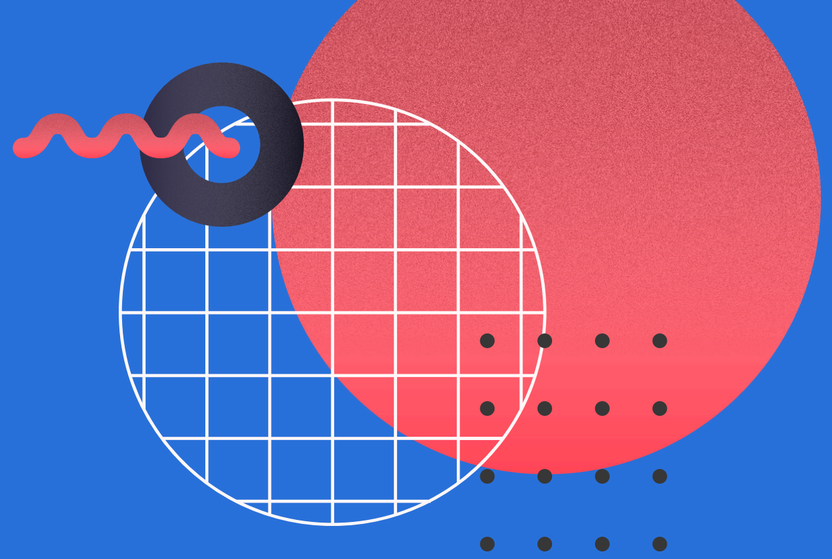


Lightweight:
SQLite is a self-contained, serverless database engine that requires minimal setup and administration. It's a great choice for embedded systems, mobile applications, or simple desktop applications where a full-fledged database server is not required.

Zero Configuration:
Since SQLite is serverless, there is no need to set up a separate database server or manage user permissions. Developers can simply include the SQLite library in their application and start using it immediately.

Cross-Platform:
SQLite is cross-platform and runs on all major operating systems, including Windows, macOS, Linux, and various mobile platforms. This makes it easy to develop applications that can be deployed across different environments without modification.

SQLite: Cons



Concurrency:
SQLite has limited support for concurrent read and write operations, especially when multiple processes or threads are accessing the database simultaneously. This can lead to performance bottlenecks in highly concurrent applications.

Scalability:
While SQLite can handle moderately sized datasets, it is not designed for high scalability or performance in distributed environments. It may struggle to keep up with the demands of large-scale applications with thousands or millions of users.

Lack of Client-Server Architecture:
Unlike traditional SQL databases, SQLite does not have a client-server architecture. This means that it cannot support multiple clients accessing the same database over a network. It's primarily intended for single-user or local use cases.

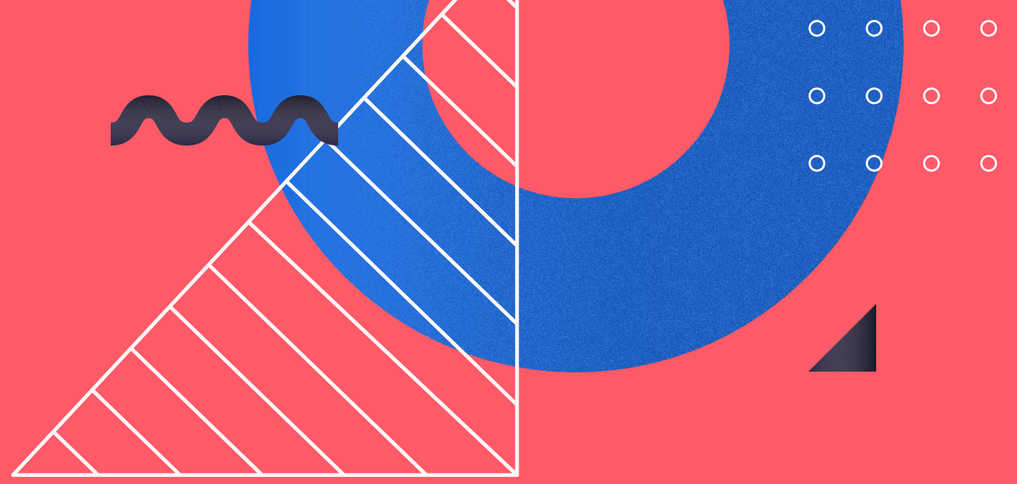
Couchbase: Pros

Scalability:
Couchbase is designed for scalability, making it suitable for large-scale applications that require high performance and availability. It can easily handle distributed data across multiple nodes.

High Availability:
Couchbase offers built-in replication and failover mechanisms, ensuring high availability and data redundancy. This makes it suitable for mission-critical applications where downtime is not acceptable.

Flexible Data Model:
Couchbase supports flexible data models, including JSON documents and key-value pairs. This flexibility allows developers to adapt to changing requirements and easily evolve their data structures over time.

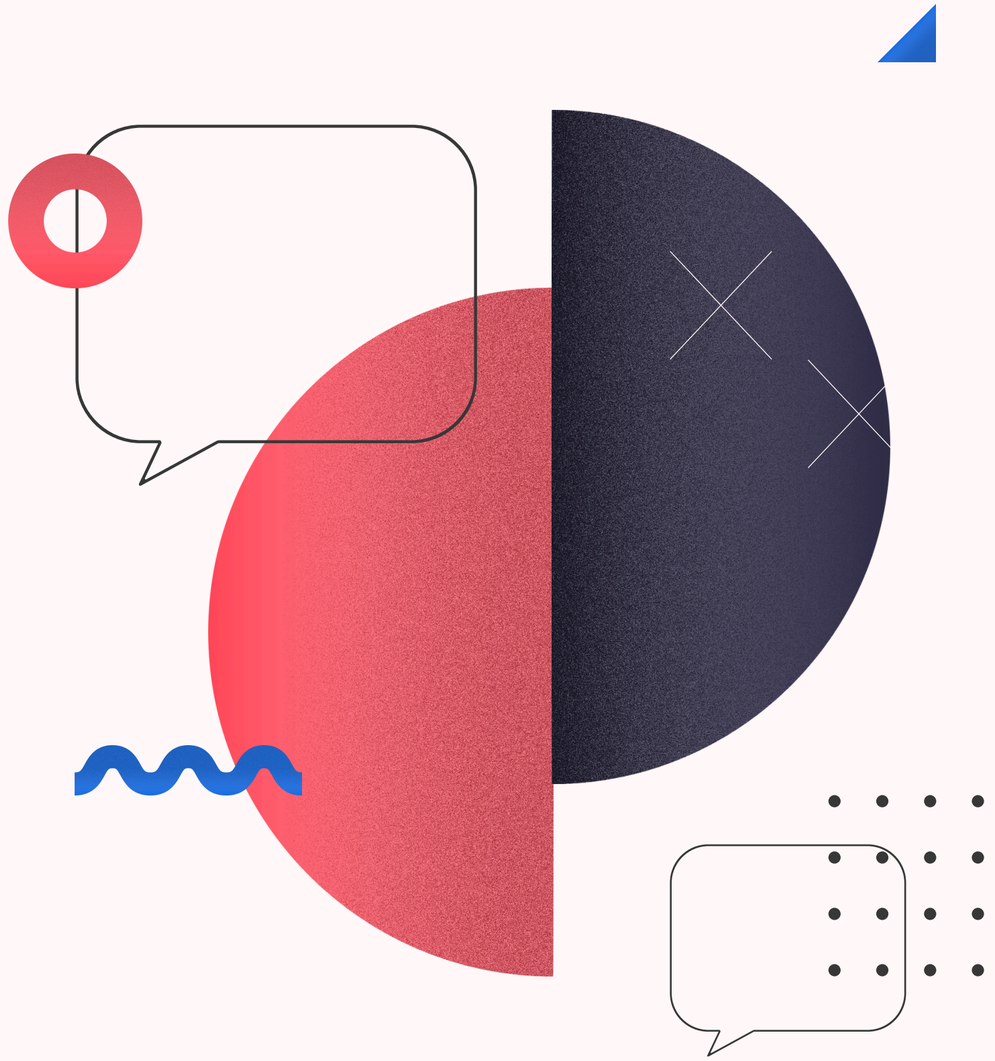
Couchbase: Cons



Complexity:
Setting up and managing a Couchbase cluster can be complex, especially for developers who are not familiar with distributed systems. It requires careful planning and configuration to ensure optimal performance and reliability.

Cost:
Couchbase is a commercial product, and while it offers a Community Edition with basic features, advanced features and support often require a paid subscription. This can make it less suitable for small or budget-constrained projects.

Learning Curve:
Couchbase has its own query language (N1QL) and administration tools, which developers need to learn in order to work effectively with the database. This learning curve can be steep for those accustomed to traditional SQL databases.



Code Samples

Thank you