

Couchbase DB

Aktuelle DB Architekturen und Technologien

Vorlesung des Studienganges Informatik
an der Dualen Hochschule Baden-Württemberg Stuttgart

von

Robert Orbach, Oliver Mohr, Oliver Braun

28.3.2025

Bearbeitungszeitraum
Matrikelnummern; Kurs
Betreuer des Dualen Partners

13.02.2025 bis 28.3.2025
8489365, xxxxxxxx, xxxxxxxx; INF22A
Prof. Dr. Carmen Winter

Inhaltsverzeichnis

Abkürzungsverzeichnis	1
1 Type of Database & History	2
1.1 Key Characteristics and Features	2
1.1.1 Data Model and Storage	2
1.1.2 Query and Indexing	2
1.1.3 Distribution and Scalability	2
1.1.4 Security and Enterprise Features	3
1.2 Brief History	3
1.3 Use Case Examples	3
2 Which aspects of relational DBs are improved?	5
2.1 Schema Flexibility	5
2.2 Scalability Improvements	5
2.3 Advanced Query Capabilities	6
2.4 Relationship to Codd's Rules	6
3 Deployment and Installation	7
3.1 Cloud deployment	7
3.1.1 How to use Couchbase Capella	7
3.2 Local installation	7
3.2.1 How to install Couchbase Server (Docker)	8
4 Deployment and Installation	9
4.1 Cloud Deployment	9
4.1.1 How to Use Couchbase Capella	9
4.2 Local Installation	9
4.2.1 How to Install Couchbase Server Using Docker	10
4.3 Choosing the Right Deployment Method	10
4.3.1 Couchbase Capella (Cloud Deployment)	10
4.3.2 Local Installation (Docker)	11
5 Advantages and Disadvantages	12
5.1 Technical Advantages	12

5.2	Technical Limitations	12
5.3	Technical Comparison	13
	Literaturverzeichnis	14
	Abbildungsverzeichnis	15

Abkürzungsverzeichnis

EUV Lithographie Extreme Ultraviolet Lithographie

1 Type of Database & History

Couchbase is a distributed NoSQL database that combines document database and key-value store capabilities. Unlike traditional single-model NoSQL solutions, Couchbase offers a hybrid approach that makes it versatile for different use cases.

1.1 Key Characteristics and Features

Couchbase distinguishes itself through several core capabilities that combine to create a versatile database platform:

1.1.1 Data Model and Storage

- **JSON Document Storage:** Schema-flexible documents supporting nested attributes, arrays, and objects (up to 20MB per document)
- **Key-Value Operations:** Sub-millisecond CRUD operations with direct key access, optimistic concurrency (CAS), and TTL expiration
- **Memory-First Architecture:** Built-in caching layer with configurable memory quotas and background disk persistence

1.1.2 Query and Indexing

- **N1QL Query Language:** SQL-like syntax for JSON with support for JOINS, nested attributes, arrays, aggregations, and subqueries
- **Comprehensive Indexing:** Primary, secondary, composite, partial, and array indexes to optimize query performance

1.1.3 Distribution and Scalability

- **Distributed Architecture:** Automatic sharding, configurable replication, auto-failover, and online rebalancing

- **Multi-Service Design:** Independent scaling of data, query, index, search, analytics, and eventing services
- **Cross-Datacenter Replication:** Built-in XDCR for geographic distribution

1.1.4 Security and Enterprise Features

- **Access Control:** Role-Based Access Control (RBAC), LDAP integration, and operation auditing
- **Encryption:** TLS for data in transit and field-level encryption for sensitive data

1.2 Brief History

Couchbase was formed in 2010 through the merger of Membase (a key-value store with Memcached compatibility) and CouchDB technology (document database capabilities). Key milestones include:

- 2011: Release of Couchbase Server 1.0
- 2015: Introduction of N1QL, a SQL-like query language for JSON
- 2017: Launch of Couchbase Mobile for edge computing
- 2021: Couchbase goes public with IPO

The introduction of N1QL in 2015 was particularly significant, as it bridged the gap between NoSQL flexibility and SQL familiarity.

1.3 Use Case Examples

Couchbase excels in several common application scenarios:

User Profile Management: Document model for dynamic attributes, fast key-value access, and session expiration.

Product Catalogs: Flexible schema for product attributes, N1QL for advanced queries,

and full-text search.

Gaming Applications: Low-latency key-value operations, document model for player/-game data, and horizontal scaling.

Internet of Things: Mobile sync for edge devices, time-series support for sensor data, and schema flexibility for device diversity.

2 Which aspects of relational DBs are improved?

Couchbase addresses three fundamental limitations of relational databases: schema rigidity, scaling challenges, and query limitations for complex data structures.

2.1 Schema Flexibility

Relational limitation: Relational databases require predefined schemas where all data must conform to a fixed structure. Schema changes typically involve ALTER TABLE operations that often cause downtime, require data migrations, and necessitate application code updates.

Couchbase solution: Couchbase's document model eliminates rigid schemas by allowing each document to have its own structure, even within the same collection. This enables:

- Adding or removing fields without database migrations
- Evolving applications without downtime
- Supporting multiple document versions simultaneously

For example, while a simple product might contain just basic attributes (id, name, price), another can include nested structures like arrays of colors and specification objects—all without schema changes.

2.2 Scalability Improvements

Relational limitation: Traditional relational databases were designed for vertical scaling (bigger servers) rather than horizontal scaling. As data grows, this approach eventually hits hardware limits. Manual sharding requires complex application logic, and

joins become problematic across distributed data.

Couchbase solution: Couchbase is architected for distributed operations from the ground up, featuring automatic sharding, built-in replication, seamless cluster expansion, and multi-dimensional scaling that allows separate scaling of query, index, and data services.

2.3 Advanced Query Capabilities

Relational limitation: SQL struggles with hierarchical data structures common in modern applications. Complex nested structures must be split across multiple tables, requiring joins that become performance bottlenecks at scale. This creates a mismatch with object-oriented application code.

Couchbase solution: N1QL (SQL for JSON) combines SQL familiarity with direct operations on nested JSON structures:

```
SELECT product.name, product.specs.cpu  
FROM products AS product  
WHERE "black" IN product.colors;
```

This query directly accesses nested fields and array elements without complex joins or subqueries.

2.4 Relationship to Codd's Rules

Couchbase intentionally diverges from relational principles to address modern application needs. While relational databases store data in rigid tables (Codd's Information Rule), Couchbase uses flexible JSON documents. This design choice prioritizes adaptability and development speed over traditional relational constraints.

3 Deployment and Installation

The Couchbase Server can be used in two different ways, depending on the deployment the user wants to achieve.

3.1 Cloud deployment

The first way is to just use Couchbase Capella, which is a cloud-based service that provides a fully managed Couchbase Server. This is the easiest way to use Couchbase, as it requires no installation or maintenance.

3.1.1 How to use Couchbase Capella

1. First, the user needs to sign up for a Couchbase Capella account. This can be done by visiting the Couchbase Capella website and following the instructions to create an account. [1]
2. Once the account has been created, the user can create a new Couchbase Server cluster by following the instructions on the Couchbase Capella website.
3. The Couchbase Server cluster should now be up and running, and the user can access the Couchbase Web Console by opening a web browser and navigating to the URL provided by Couchbase Capella.

3.2 Local installation

The second way is to install the Couchbase Server locally. It is best to use Docker for this, even though it is a more complex way to use Couchbase, as it requires the user to have Docker installed on their machine. However, this method provides more flexibility and control over the Couchbase Server.

3.2.1 How to install Couchbase Server (Docker)

1. First, the user needs to install Docker on their machine. This can be done by following the instructions on the Docker website.
2. Next, the user needs to pull the Couchbase Server image from the Docker Hub. This can be done by running the following command in the terminal:

```
docker pull couchbase/server
```

3. Once the image has been pulled, the user can run the Couchbase Server container by running the following command in the terminal:

```
docker run -d --name couchbase -p 8091-8094:8091-8094 -p 11210:11210 couchbase
```

4. The Couchbase Server should now be running on the user's machine. The user can access the Couchbase Web Console by opening a web browser and navigating to <http://localhost:8091>.
-

4 Deployment and Installation

Couchbase Server can be deployed in two different ways, depending on the user's needs and infrastructure preferences.

4.1 Cloud Deployment

The simplest way to use Couchbase Server is through Couchbase Capella, a fully managed cloud-based service. This eliminates the need for local installation and ongoing maintenance, making it an excellent choice for users who prefer a hassle-free, scalable solution.

4.1.1 How to Use Couchbase Capella

To get started with Couchbase Capella, follow these steps:

1. Sign up for a Couchbase Capella account by visiting the official Couchbase Capella website and following the registration instructions. [1]
2. Once the account is created, set up a new Couchbase Server cluster by following the step-by-step guide provided within the Capella interface.
3. After the cluster is successfully deployed, access the Couchbase Web Console by opening a web browser and navigating to the URL provided by Couchbase Capella.

4.2 Local Installation

For users who prefer greater control over their deployment, installing Couchbase Server locally is a viable option. The recommended approach for this is using Docker,

which simplifies the setup process while offering flexibility. However, this method requires Docker to be installed on the machine and some level of system configuration.

4.2.1 How to Install Couchbase Server Using Docker

To install Couchbase Server with Docker, follow these steps:

1. Install Docker on your machine by following the official installation guide available on the Docker website.
2. Pull the Couchbase Server image from Docker Hub by running the following command in the terminal:

```
docker pull couchbase/server
```

3. Run the Couchbase Server container with the following command:

```
docker run -d --name couchbase -p 8091-8094:8091-8094 -p 11210:11210 couchbase/server
```

4. Once the container is running, access the Couchbase Web Console by opening a web browser and navigating to <http://localhost:8091>.

4.3 Choosing the Right Deployment Method

Selecting between Couchbase Capella and a local installation depends on various factors. Below is a comparison to help determine the best approach:

4.3.1 Couchbase Capella (Cloud Deployment)

Advantages:

- Fully managed service with automated updates and backups.
- High availability and scalability.
- No infrastructure maintenance required.

Disadvantages:

- Ongoing costs for cloud usage.
- Less flexibility in configuration and customization.
- Internet dependency for access.

4.3.2 Local Installation (Docker)

Advantages:

- Full control over configuration and security settings.
- No cloud-related costs.
- Suitable for local development and testing.

Disadvantages:

- Requires manual maintenance, updates, and backups.
- Limited by local machine resources.
- More complex initial setup compared to Capella.

In summary, Couchbase Capella is ideal for businesses and teams looking for a fully managed, scalable solution, while a local installation using Docker is best suited for developers and users requiring complete control over their environment.

5 Advantages and Disadvantages

5.1 Technical Advantages

Performance Metrics:

- Sub-millisecond key-value operations (<1ms @ 99th percentile)
- Memory-first architecture with configurable ejection policies
- B-tree based global secondary indexes for query optimization
- Write-optimized storage engine with append-only commits
- 30-40

Technical Capabilities:

- Multi-model support: K-V, document, spatial, full-text within single platform
- ANSI JOIN and NEST operations in N1QL with pushdown optimization
- Cross Datacenter Replication (XDCR) with filtering and compression
- SSLv3/TLS 1.2+ encryption with FIPS 140-2 compliance
- SDK support for Java, .NET, Node.js, Python, Go with reactive extensions

5.2 Technical Limitations

Performance Constraints:

- Document size limit: 20MB (default)
- Memory overhead: 56 bytes metadata per document
- Transaction latency: increased by 15-30% for multi-document ACID
- Query performance degrades with >10 JOINS in single statement
- Minimum 4GB RAM recommended per node for production

Architectural Considerations:

- Default consistency: eventual for queries, strong for K-V operations
- CAP theorem positioning: CP for document operations, AP for cross-cluster
- Minimum 3 nodes recommended for high availability
- Scaling requires rebalance operations (minimal but measurable impact)
- Analytics segregation required for OLAP without OLTP impact

5.3 Technical Comparison

Feature	Couchbase	MongoDB	Cassandra
Query Latency	1-5ms K-V, 5-20ms N1QL	2-10ms K-V, 10-50ms query	1-3ms K-V, 5-15ms CQL
Scaling Model	Shared-nothing, auto-sharding	Replica sets with sharding	Masterless ring
Consistency	Tunable (BASE to ACID)	Tunable (Read preferences)	Quorum-based
Indexes	Memory-optimized, composite	B-tree, compound	LSM-tree, materialized views
Transactions	ACID within/a-cross docs	ACID within/a-cross docs	Lightweight transactions

Ideal Workloads: High-throughput OLTP with sub-ms requirements; mixed document/K-V operations; distributed multi-model applications requiring SQL compatibility.

Less Suitable: Graph-centric applications (high relationship traversal); pure analytics workloads; single-server deployments with <8GB RAM.

Literaturverzeichnis

- [1] „Mariadb Documentation.“ (o.J.), Adresse: <https://cloud.couchbase.com/signup> (besucht am 04.03.2025).

Abbildungsverzeichnis