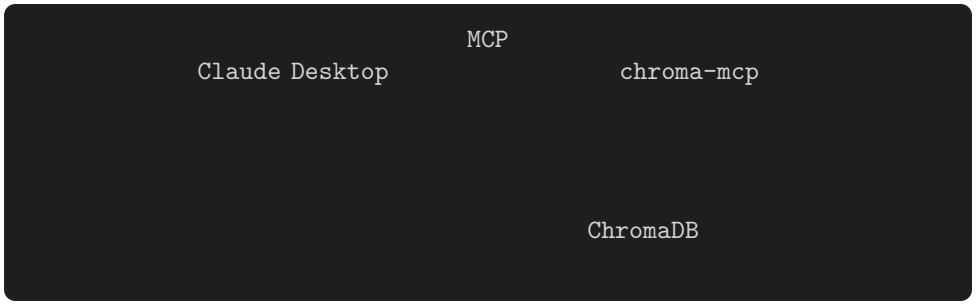
ChromaDB MCP-Server

Das Handbuch - aus der 'Dr.Jo's dive in' Reihe

Lokale Vektor-Datenbank für Claude Desktop

F.R. Granaria Publishers



Version 1.0

31. Januar 2026

Inhaltsverzeichnis

1 Übersicht

1.1 Was ist ChromaDB?

ChromaDB ist eine Open-Source Vektor-Datenbank, die **semantische Suche** ermöglicht. Anstatt nach exakten Wörtern zu suchen, findet ChromaDB Dokumente basierend auf ihrer **Bedeutung**.

- Suche nach ``Wie beantrage ich Urlaub?'' findet auch ``Freie Tage müssen 2 Wochen vorher eingereicht werden''
- Verwendet Vektor-Embeddings zur Bedeutungserfassung
- Millisekunden-Antwortzeiten auch bei großen Datenmengen

1.2 Was ist der MCP-Server?

Das **Model Context Protocol (MCP)** ermöglicht Claude Desktop direkten Zugriff auf externe Tools und Datenquellen. Der ChromaDB MCP-Server verbindet Claude mit deiner lokalen Vektor-Datenbank.

1.3 Vorteile

Vorteil	Beschreibung
Semantische Suche	Finde relevante Inhalte nach Bedeutung
Lokal & Privat	Alle Daten bleiben auf deinem Rechner
Persistent	Daten überleben Neustarts
¤ Schnell	Millisekunden-Antwortzeiten
Kostenlos	Keine API-Kosten (lokales Embedding-Modell)

2 Installation

2.1 Voraussetzungen

- Python 3.11 oder höher
- Claude Desktop App
- macOS / Linux / Windows

2.2 Schritt 1: ChromaDB und MCP-Server installieren

```
# ChromaDB installieren
pip3 install chromadb --break-system-packages

# MCP-Server installieren
pip3 install chroma-mcp --break-system-packages
```

2.3 Schritt 2: Installation verifizieren

```
# Pfad zum Executable finden
which chroma-mcp
# Ausgabe: /Library/Frameworks/Python.framework/Versions/3.11/bin/chroma-
    mcp

# Hilfe anzeigen
chroma-mcp --help
```

2.4 Schritt 3: Datenverzeichnis erstellen

```
mkdir -p ~/clawd/chromadb
```

3 Konfiguration

3.1 Claude Desktop Konfigurationsdatei

Die Konfigurationsdatei befindet sich unter:

Betriebssystem	Pfad
macOS	`~/Library/Application Support/Claude/claude_desktop_config.json`
Windows	`%APPDATA%\Claude\claude_desktop_config.json`
Linux	`~/.config/Claude/claude_desktop_config.json`

3.2 MCP-Server Eintrag hinzufügen

Öffne die Konfigurationsdatei und füge den `chroma` Server hinzu:

```
{
  "mcpServers": {
    "chroma": {
      "command": "/Library/Frameworks/Python.framework/Versions/3.11/bin/
          chroma-mcp",
      "args": [
        "--client-type", "persistent",
        "--data-dir", "/Users/DEIN_USERNAME/clawd/chromadb"
      ]
    }
  }
}
```

© Warnung

Ersetze `/Users/DEIN_USERNAME/` mit deinem tatsächlichen Benutzerpfad!

3.3 Client-Typen

Typ	Verwendung	Beschreibung
`persistent`	**Empfohlen**	Daten werden auf Festplatte gespeichert
`ephemeral`	Testing	Nur im RAM, geht bei Neustart verloren
`http`	Server-Setup	Verbindung zu externem ChromaDB Server
`cloud`	Enterprise	Chroma Cloud Service

3.4 Konfiguration aktivieren

Nach dem Speichern der Konfiguration:

1. Claude Desktop **komplett beenden** (nicht nur Fenster schließen)
2. Claude Desktop **neu starten**
3. Neuen Chat öffnen

4 MCP-Tools Referenz

4.1 Übersicht aller Tools

Tool	Funktion
`chroma_list_collections`	Alle Collections auflisten
`chroma_create_collection`	Neue Collection erstellen
`chroma_get_collection_info`	Collection-Details abrufen
`chroma_get_collection_count`	Anzahl Dokumente zählen
`chroma_peek_collection`	Erste Dokumente anzeigen
`chroma_modify_collection`	Collection umbenennen/bearbeiten
`chroma_delete_collection`	Collection löschen
`chroma_add_documents`	Dokumente hinzufügen
`chroma_query_documents`	Semantische Suche
`chroma_get_documents`	Dokumente nach ID/Filter abrufen
`chroma_update_documents`	Dokumente aktualisieren
`chroma_delete_documents`	Dokumente löschen

4.2 Collection Management

4.2.1 chroma_list_collections

Listet alle vorhandenen Collections auf.

```
chroma_list_collections()
chroma_list_collections(limit=10, offset=0)
```

Parameter:

- `limit` (optional): Maximale Anzahl zurückzugebender Collections

- `offset` (optional): Überspringe die ersten N Collections

4.2.2 chroma_create_collection

Erstellt eine neue Collection.

```
chroma_create_collection(
    collection_name="meine_collection",
    embedding_function_name="default",
    metadata={"description": "Meine Beschreibung"}
)
```

Parameter:

- `collection_name` (erforderlich): Name der Collection
- `embedding_function_name` (optional): Embedding-Modell
  - `default` -- all-MiniLM-L6-v2 (lokal, kostenlos)
  - `openai` -- OpenAI Embeddings (API-Key nötig)
  - `cohere` -- Cohere Embeddings (API-Key nötig)
- `metadata` (optional): Zusätzliche Metadaten

4.2.3 chroma_delete_collection

Löscht eine Collection **unwiderruflich**.

```
chroma_delete_collection(collection_name="test_collection")
```

Achtung

Diese Aktion kann nicht rückgängig gemacht werden! Alle Dokumente in der Collection werden gelöscht.

4.3 Dokument-Operationen

4.3.1 chroma_add_documents

Fügt neue Dokumente hinzu.

```
chroma_add_documents(
    collection_name="prompts_trading",
    documents=[
        "Analysiere den BTC/USD Chart mit RSI und MACD",
        "Erstelle eine Watchlist fuer Tech-Aktien"
    ],
    ids=[
        "trading_001",
        "trading_002"
    ],
    metadatas=[
        {"name": "BTC Analyse", "tags": "crypto,technical"},
        {"name": "Tech Watchlist", "tags": "stocks,watchlist"}
    ]
)
```

Parameter:

- `collection_name` (erforderlich): Ziel-Collection
- `documents` (erforderlich): Liste der Texte
- `ids` (erforderlich): Eindeutige IDs für jedes Dokument
- `metadatas` (optional): Liste von Metadaten-Dictionaries

4.3.2 chroma_query_documents

Semantische Suche -- das Herzstück von ChromaDB!

```
chroma_query_documents(
    collection_name="prompts_analysis",
    query_texts=["competitor strategy"],
    n_results=5,
    where={"rating": {"$gte": 4}},
    include=["documents", "metadatas", "distances"]
)
```

Parameter:

- `collection_name` (erforderlich): Zu durchsuchende Collection
- `query_texts` (erforderlich): Liste von Suchanfragen
- `n_results` (optional): Anzahl Ergebnisse (default: 5)
- `where` (optional): Metadaten-Filter
- `where_document` (optional): Dokument-Inhaltsfilter
- `include` (optional): Was zurückgegeben werden soll

4.4 Filter-Operatoren

4.4.1 Metadaten-Filter (where)

Operator	Bedeutung	Beispiel
`$eq`	Gleich	`{ßtatus": {"$eq": äctive"}}`
`$ne`	Ungleich	`{ßtatus": {"$ne": "deleted"}}`
`$gt`	Größer als	`{"rating": {"$gt": 3}}`
`$gte`	Größer oder gleich	`{"rating": {"$gte": 4}}`
`$lt`	Kleiner als	`{"count": {"$lt": 100}}`
`$lte`	Kleiner oder gleich	`{"count": {"$lte": 50}}`
`$in`	In Liste	`{"theme": {"$in": [ä", "b"]}}`
`$nin`	Nicht in Liste	`{"theme": {"$nin": ["test"]}}`

4.4.2 Logische Operatoren

AND-Verknüpfung:

```
{
    "$and": [
        {"rating": {"$gte": 4}},
        {"theme": {"$eq": "trading"}}
    ]
}
```

OR-Verknüpfung:

```
{
    "$or": [
        {"theme": {"$eq": "trading"}},
        {"theme": {"$eq": "analysis"}}
    ]
}
```

4.4.3 Dokument-Filter (where_document)

Operator	Bedeutung	Beispiel
`$contains`	Enthält Text	`{"$contains": "Python"}`
`$not_contains`	Enthält nicht	`{"$not_contains": "Java"}`

5 Praktische Beispiele

5.1 Beispiel 1: Wissensdatenbank aufbauen

```
# Collection erstellen
chroma_create_collection(
    collection_name="company_knowledge",
    metadata={"description": "Firmenwissen und Prozesse"}
)

# Dokumente hinzufuegen
chroma_add_documents(
    collection_name="company_knowledge",
    documents=[
        "Urlaubsantraege muessen 2 Wochen vorher eingereicht werden.",
        "Die IT-Hotline ist erreichbar unter 555-1234.",
        "Spesenabrechnung erfolgt ueber das SAP-Portal."
    ],
    ids=["hr_001", "it_001", "finance_001"],
    metadatas=[
        {"department": "HR", "topic": "urlaub"},
        {"department": "IT", "topic": "support"},
        {"department": "Finance", "topic": "spesen"}
    ]
)

# Semantische Suche
chroma_query_documents(
    collection_name="company_knowledge",
```

```
    query_texts=["Wie beantrage ich frei?"],
    n_results=3
)
```

Tipp

Die Suche nach ``Wie beantrage ich frei?'' findet auch Dokumente über ``Urlaubsanträge'' -- das ist die Stärke der semantischen Suche!

5.2 Beispiel 2: Code-Snippets verwalten

```
chroma_create_collection(collection_name="code_snippets")

chroma_add_documents(
    collection_name="code_snippets",
    documents=[
        "def read_csv(path): return pd.read_csv(path)",
        "async def fetch_api(url): async with aiohttp.get(url) as r:
            return await r.json()",
        "SELECT * FROM users WHERE created_at > NOW() - INTERVAL '7 days'"
    ],
    ids=["py_csv", "py_async", "sql_recent"],
    metadatas=[
        {"language": "python", "category": "io"},
        {"language": "python", "category": "async"},
        {"language": "sql", "category": "query"}
    ]
)

# Nur Python-Snippets suchen
chroma_query_documents(
    collection_name="code_snippets",
    query_texts=["Datei einlesen"],
    where={"language": {"$eq": "python"}}
)
```

6 Prompt-Vault Integration

Das Prompt-Vault System nutzt ChromaDB als Backend für die Speicherung und Suche von Prompts.

6.1 Collections

Collection	Thema	Icon
`prompts_trading`	Trading & Finanzen	
`prompts_coding`	Programmierung	
`prompts_writing`	Texte & Dokumente	Ǝ
`prompts_analysis`	Analysen & Research	
`prompts_creative`	Kreative Inhalte	
`prompts_system`	System-Prompts	ˆ
`prompts_general`	Allgemeines	
`prompts_ecommerce`	E-Commerce	
`prompts_marketing`	Marketing	

6.2 CLI-Tool pv

Zusätzlich zum MCP gibt es das Kommandozeilen-Tool:

```
pv list                    # Alle Prompts anzeigen
pv list -t trading         # Nach Thema filtern
pv search "competitor"     # Semantische Suche
pv show <id>               # Prompt-Details anzeigen
pv add -t marketing        # Neuen Prompt hinzufuegen
pv stats                   # Statistiken
```

7 Troubleshooting

7.1 Problem: ``No result received from client-side tool execution''

Ursache: MCP-Server nicht verbunden

Lösung:

1. Claude Desktop komplett beenden
2. Im Terminal prüfen: `ps aux | grep -i claude`
3. Falls noch Prozesse laufen: `killall Claude`
4. Claude Desktop neu starten

7.2 Problem: ``command not found: chroma-mcp''

Ursache: Pfad nicht korrekt in der Konfiguration

Lösung:

```
# Korrekten Pfad finden
which chroma-mcp
# oder
find /Library -name "chroma-mcp" 2>/dev/null

# Diesen Pfad in der Config verwenden
```

7.3 Problem: Collection existiert nicht

Ursache: Falscher Datenbank-Pfad

Lösung:

```
# Pruefen ob Daten vorhanden
ls -la ~/clawd/chromadb/

# Config pruefen
cat ~/Library/Application\ Support/Claude/claude_desktop_config.json
```

8 Best Practices

8.1 Naming Conventions

- **Collections:** `snake_case`, Präfix nach Typ (`prompts_`, `docs_`, `notes_`)
- **IDs:** `{theme}_{timestamp}` oder `{theme}_{sequential}`
- **Metadaten-Keys:** Konsistent über alle Dokumente

8.2 Metadaten-Schema

Definiere ein konsistentes Schema für alle Dokumente:

```
{
    "name": "Kurzer Titel",
    "theme": "hauptkategorie",
    "subcategory": "unterkategorie",
    "tags": "komma,getrennte,tags",
    "rating": 5,
    "source": "URL oder Quelle",
    "created": "2026-01-31T14:00:00",
    "updated": "2026-01-31T14:00:00"
}
```

8.3 Performance-Tipps

1. **Batch-Inserts:** Füge viele Dokumente in einem Call hinzu
2. **Spezifische Queries:** Nutze `where`-Filter um Ergebnismenge einzuschränken
3. **n_results begrenzen:** Nicht mehr anfordern als nötig
4. **Collections aufteilen:** Thematisch trennen statt eine riesige Collection

8.4 Backup

```
# Einfaches Backup
cp -r ~/clawd/chromadb ~/clawd/chromadb_backup_$(date +%Y%m%d)

# Mit rsync (inkrementell)
rsync -av ~/clawd/chromadb/ ~/Backups/chromadb/
```

Anhang: Schnellreferenz

Häufigste Befehle

Aktion	Befehl
Collections anzeigen	`chroma_list_collections()`
Dokumente zählen	`chroma_get_collection_count(collection_name="X")`
Suchen	`chroma_query_documents(collection_name="X", query_texts=["..."])`
Hinzufügen	`chroma_add_documents(collection_name="X", documents=[...], ids=[...])`
Löschen	`chroma_delete_documents(collection_name="X", ids=[...])`

Embedding-Modelle

Name	Dimensionen	Lokal	Kosten
default (MiniLM)	384		Kostenlos
openai	1536	~	~$0.0001/1k tokens
cohere	1024	~	~$0.0001/1k tokens

Ressourcen

- ChromaDB Dokumentation: https://docs.trychroma.com/
- MCP Spezifikation: https://modelcontextprotocol.io/
- chroma-mcp GitHub: https://github.com/chroma-core/chroma-mcp