



Studienarbeit - Entwurf

Ausarbeitung InfluxDB und Wetterdaten

Erstellt von:

Henrik Mertens
Hatzfelder Str 25
33104 Paderborn

Prüfer:

Prof. Dr. Ulrich. Reus

Eingereicht am:

24. Juni 2022

Inhaltsverzeichnis

Abkürzungsverzeichnis	IV
Abbildungsverzeichnis	V
Tabellenverzeichnis	VI
Listingverzeichnis	VII
1 Einleitung	1
1.1 Zielsetzung	1
1.2 Aufbau und Vorgehensweise	1
2 Grundlagen	2
2.1 Time Series Data	2
2.2 Aufbau von Time Series Datenbanken	3
2.3 Unterschiede zwischen Time Series und relationalen Datenbanken	3
2.4 Verbreitete DBMS	4
2.5 Entwicklungsumgebung	4
2.5.1 Docker und Docker-Compose	5
2.5.2 Python und Jupyter Notebooks	5
3 InfluxDB	7
3.1 InfluxDB Installation	7
3.2 Daten einfügen	8
3.3 Line Protokoll	8
3.4 Daten abrufen und visualisieren	9
3.5 Python library	10
3.6 Wetterdaten	11
3.6.1 Wetterdaten Aufbau	11
3.6.2 Wetterdaten abrufen	14
3.6.3 Wetterdaten Verarbeiten	14
4 Kritische Auseinandersetzung	15
5 Zusammenfassung	15

6 Schlussfolgerung und Fazit	15
Anhang	16
Quellenverzeichnis	25
Ehrenwörtliche Erklärung	28

Abkürzungsverzeichnis

API	Application Programming Interface.
CLI	Command Line Interface.
CRUD	Create, Read, Update, Delete.
CSV	Comma-separated values.
DWD	Deutscher Wetterdienst.
HTML	Hypertext Markup Language.
HTTP	HyperText Transfer Protocol.
IOT	Internet of Things.
RDBMS	Relational Database Management System.
TSDB	Time Series Database.
UI	User Interface.
URL	Uniform Resource Locator.

Abbildungsverzeichnis

Abbildung 1: DB-Engines Ranking	18
Abbildung 2: InfluxDB Dashboard	20
Abbildung 3: InfluxDB Load Data Source	21
Abbildung 4: InfluxDB Load Data Bucket	22
Abbildung 5: InfluxDB Load Data Bucket hinzufügen	22
Abbildung 6: InfluxDB Load Data API Tokens	23
Abbildung 7: InfluxDB Load Data API Token hinzufügen	24

Tabellenverzeichnis

Tabelle 1: Bedeutung der CSV Felder	13
Tabelle 2: Bedeutung Stationstabellen Felder	13

Listingverzeichnis

Listing 1: InfluxDB Line Protokoll Quelle: Influxdata (2022c)	8
Listing 2: Wetterdaten CSV Quelle: DWD Wetterdaten CSV Download	12
Listing 3: InfluxDB Python Daten schreiben Quelle: Influxdata (2022e)	17
Listing 4: InfluxDB Python Daten lesen Quelle: Influxdata (2022e)	17
Listing 5: InfluxDB Flux Beispiel Quelle: Influxdata (2022e)	19

1 Einleitung

1.1 Zielsetzung

Das Ziel dieser Arbeit ist es eine Einführung in die Funktion von Time Series Database (TSDB) zu geben. Dazu wird beispielhaft an den Temperaturdaten des Deutschen Wetterdienstes (DWDs) gezeigt wie einer TSDB gearbeitet werden kann. Als Datenbank wird InfluxDB genutzt.

1.2 Aufbau und Vorgehensweise

Im Grundlagenteil dieser Arbeit werden die Grundlagen von Time Series Data und TSDB erläutert. Des Weiteren wird erklärt was eine TSDB ist und wie diese sich von Relational Database Management System (RDBMS) unterscheidet. Außerdem wird kurz die für die Entwicklungsumgebung notwendige Software Docker und jupyter Notebooks erklärt.

Im nächsten Kapitel wird gezeigt wie mit InfluxDB gearbeitet werden kann. Dazu wird das für InfluxDB wichtige Line Protokoll vorgestellt. Außerdem wird gezeigt wie die abgerufenen Daten mit InfluxDB visualisiert werden können. Als Daten werden die Wetterdaten des DWD genutzt welche auch in diesem Kapitel erklärt und vorgestellt werden.

Im letzten Artikel folgt eine kurze Zusammenfassung der Arbeit und der wichtigsten Informationen zu TSDB

2 Grundlagen

In diesem Kapitel werden die Grundlagen von TSDB und Time Series Data erklärt. TSDB gehören zu den NoSQL Datenbanken und sind besonders darauf optimiert, mit Time Series Data zu arbeiten. Dadurch können die große Mengen an Time Series Data verarbeiten, durchsuchen und speichern.¹

2.1 Time Series Data

Um TSDB zu verstehen, muss als erstes geklärt werden, was Time Series Data überhaupt ist und wie sie sich von anderen Daten unterscheiden. Wie der Name schon impliziert, ist Time Series Data eine Reihe von Daten, die über einen Zeitraum gesammelt worden sind. Es wird also nicht nur der Endwert aufgezeichnet, sondern die Veränderung über einen Zeitraum. Diese Daten können z.B. Servermetriken, Netzwerkdaten, Internet of Things (IOT) Sensordaten, Ereignisse, Klicks, Marktgeschäfte und viele andere Arten von Daten sein. Time Series Data können gut daran erkannt werden, dass die Zeit eine wichtige Achse bei der Darstellung der Werte ist.²

Manchmal ist es nicht notwendig, alle Daten zu erfassen. Zum Beispiel wird in vielen Anwendungen nur der letzte Login gespeichert. Mehr ist auch für die Funktion nicht notwendig. Allerdings können zusätzliche Informationen gewonnen werden, wenn nicht nur der letzte Datenpunkt sondern die Veränderung über einen Zeitraum aufgezeichnet wird. So kann zum Beispiel festgestellt werden, wie oft und wann sich der Kunde einloggt und ob es dabei ein Muster gibt. Anhand dieser Daten können Kunden dann kategorisiert werden.³

Eine weitere Besonderheit von Time Series Data ist, dass sie sich nicht verändert. Wenn die Daten einmal erfasst wurden wird an ihnen nichts mehr verändert. Es werden nur neue Daten hinzugefügt.⁴

¹vgl. ComputerWeekly.de, Redaktion (2021)

²vgl. Dix, Paul (2021), S. 1 ff.

³vgl. Data-Science-Team (2020)

⁴vgl. Fangman, Sam (2019)

2.2 Aufbau von Time Series Datenbanken

TSDB sind darauf ausgelegt Key Value paare zu Speichern. Der Key in einem Key Value Datensatz ist eine Wert über den die Value referenziert wird. Im Value Teil werden die Daten zum dazugehörigem Key gespeichert. Der Wert der Value kann ein primitiver Datentyp sein oder auch ein Objekt das in einen primitiven Datentyp umgewandelt worden ist.⁵

Ein Datenpunkt in einer Time Series Database besteht aus mehreren key Value Paaren. Einige Dieser Key Value Paare sind sogenannte Tags. Diese Tags sind Werte die sich zwischen den Datenpunkten nicht ändern wie zum Beispiel die Position eines Sensors oder die Kundennummer eines Kunden der sich gerade eingeloggt hat. Anhand dieser Tags können die Datenpunkte durchsucht werden. Die eigentlichen Messwerte des Sensors oder andere Daten die erfasst werden sollen werden auch als Key Value Paar gespeichert. Zum Beispiel wird als Name Temperatur und als Wert 25,2 angegeben. Ein Datenpunkt kann mehrere Messwerte haben. Außerdem wird jeder Datenpunkt mit einem Timestamp versehen nach welchem er Indexiert wird.⁶ Eine gute Veranschaulichung wie die Daten in einer TSDB aufgebaut sind zeigt das InfluxDB Line Protokoll in Listing 1.

2.3 Unterschiede zwischen Time Series und relationalen Datenbanken

Um Time Series Data zu speichern, ist es nicht unbedingt erforderlich, eine TSDB zu nutzen. Auch relationale Datenbanken können Time Series Data speichern. Einer der wichtigsten Unterschiede zwischen einer TSDB im Gegensatz zu einem RDBMS ist es, dass kein Datenbank Schema benötigt wird. Wenn Time Series Daten in einer Relationalen Datenbank geschrieben werden sollen, müssen erst entsprechende Tabellen angelegt werden, in denen die Daten immer im gleichen Format abgelegt werden müssen. Im Gegensatz dazu können in einer TSDB die Daten einfach schemafrei in die Datenbank geschrieben werden. Ein weiterer Vorteil ist, dass TSDB im Gegensatz zu relationalen Datenbanken besser und einfacher Skaliert werden können.⁷

⁵vgl. Seeger, Marc (2009)

⁶vgl. hazelcast (2022)

⁷vgl. Influxdata (2021c)

Aber TSDB haben nicht nur Vorteile. Wie in Abb. 1 zu sehen, sind sie viel weniger verbreitet als nicht zeit basierte Datenbank Systeme. Dadurch gibt es viel weniger Entwickler die Erfahrungen mit TSDB haben. Auch das Ökosystem um die Datenbank ist deutlich kleiner. Außerdem sind RDBMS dadurch, dass es sie viel länger gibt, sehr stabil und sehr gut unterstützt.⁸

RDBMS arbeiten nach dem Create, Read, Update, Delete (CRUD) Prinzip, welches für Time Series Data nicht optimal ist. Auf Time Series Data werden keine Update Befehle durchgeführt, da neue Daten immer nur angehängt werden. Auch das Löschen von Daten wird nicht sehr häufig durchgeführt und im Gegensatz zu RDBMS meistens gleichzeitig auf einer großen Menge an Datensätzen. TSDB sind auf diese Besonderheiten optimiert und daher besser dafür geeignet, mit Time Series Data zu arbeiten und weisen auch eine höhere Performance auf.⁹

2.4 Verbreitete DBMS

Aktuell gibt es wie in Abb. 1 zu sehen, einige beliebte Multi-Model Datenbanken, die als TSDB genutzt werden können. So können die Datenbanken wie MongoDB, Redis, Teradata und Couchbase mit Time Series Daten arbeiten. Die erste reine TSDB im Ranking ist InfluxDB auf Platz 29.¹⁰

Allerdings haben Datenbanken, die nur auf das Verarbeiten von Time Series Data ausgelegt sind, deutliche Performance Vorteile gegenüber Multi Model Datenbanken. In einem Vergleich von InfluxDB und MongoDB, hat InfluxDB eine 2,4 mal bessere Schreibperformance als MongoDB und ist beim Lesen sogar 5,7 mal schneller. InfluxDB benötigt außerdem 20 mal weniger Speicherplatz auf der Festplatte, um die gleiche Menge an Daten zu speichern.¹¹

2.5 Entwicklungsumgebung

Um mit InfluxDB zu arbeiten wird eine Umgebung zum Ausführen von Docker Containern benötigt, in welchen wir InfluxDB und Jupyter Notebooks betreiben. Der

⁸vgl. Influxdata (2021c)

⁹vgl. Influxdata (2021c)

¹⁰vgl. solid-IT-gmbh (2022)

¹¹vgl. Hajek Vlasta Pour Ales, Kudibal Ivan (2019)

eigentliche Code wird dann in Jupyter Notebooks mit Python entwickelt. Die Grundlagen über die eingesetzten Tool und Techniken werden grob in diesem Kapitel erläutert.

2.5.1 Docker und Docker-Compose

Docker ist eine Software für das erstellen und verwalten von Containern. Mit Docker ist es möglich Anwendungen samt ihrer Umgebung in einer Einheit zusammenzufassen, so dass diese einfach auf anderen System ausgeführt werden können. Dabei hat jeder Container ein eigenes Dateisystem und ein eigenes Betriebssystem. Allerdings teilen sich Container und Hostsystem den Kernel des Hostsystems. Dadurch hat diese Art der Virtualisierung deutlich weniger Overhead als andere Virtualisierungstechniken. Zusätzlich wird das Betriebssystem innerhalb des Containers maximal reduziert so dass nur noch benötigte Komponenten vorhanden sind. Wichtig ist es dass immer nur möglichst eine Anwendung in einem Container zu finden ist. Durch die Virtualisierung sind die einzelnen Container voneinander getrennt.¹²

Allerdings bestehen einige Anwendungen aus mehreren Komponenten, diese können durch mehrere Docker Container abgebildet werden. Um die Verwaltung von mehreren Containern zu erleichtern kann Docker-Compose genutzt werden. Mithilfe von Docker Compose können größere Umgebungen in einem Compose File verwaltet werden. Hier werden die Umgebungsvariablen, Container Image oder Dockerfiles, Ports, Storage und weiteres in einer Datei definiert. Mithilfe dieser Definition kann Docker Compose eine komplexe Umgebung mit nur einem Befehl initialisieren.¹³

2.5.2 Python und Jupyter Notebooks

Python ist eine universelle Prozedurale und Imperative Programmiersprache die 1994 in der ersten Version veröffentlicht wurde. Der Name ist eine Huldigung an Monty Python und wurde nicht nach einer Schlange benannt, auch wenn das Logo eine Schlange ist. Python ist unter der freien PFS Lizenz lizenziert wodurch es auch in kommerziellen Anwendung genutzt werden kann. Python ist eine Interpretierte Sprache. Das heißt dass sie nicht zu einer ausführbaren Datei kompiliert wird sondern von einem Interpreter

¹²vgl. Stender, Daniel (2020), S. 54 ff.

¹³vgl. Stender, Daniel (2020), S. 151 ff.

interpretiert wird. Außerdem ist Python eine unter Programmieranfängern sehr beliebte Sprache die auch sehr viel in den Bereichen DataScience, DeepLearning, Naturwissenschaften und Linux Systemprogrammierung eingesetzt wird.¹⁴

Jupyter Notebooks ist eine Webbasierte Open-Source Anwendung mit dem Ziel Code in den Sprachen Python, R, und Julia einfach zu schreiben, bearbeiten, auszuführen und einfach zu teilen. Ein Notebook besteht immer aus Zellen. Eine Zelle kann Code oder Markdown Formatierten Text anzeigen. Jede Zelle kann einzeln ausgeführt werden. Dadurch kann ein Programm sehr einfach und verständlich dargestellt und erklärt werden. Es ist auch möglich neue Notebooks und Dateien im Webinterface von Jupyter Notebooks selbst anzulegen.¹⁵

¹⁴vgl. Stender, Daniel (2020), S. 66 ff.

¹⁵vgl. Silaparasetty, Nikita (2020), S. 91 ff.

3 InfluxDB

InfluxDB ist eine in Go geschriebene open source TSDB, die darauf ausgelegt ist, mit einer großen Menge an Time Series Data zu arbeiten.¹⁶ Im weiteren Verlauf dieses Kapitels wird am Beispiel von Wetterdaten gezeigt, wie mit InfluxDB gearbeitet wird. InfluxDB stellt für die Integration in eigene Anwendungen ein HyperText Transfer Protocol (HTTP) Application Programming Interface (API) zur Verfügung, für die es in vielen Programmiersprachen Client Librarys gibt. Außerdem wird ein Webinterface und ein Command Line Interface (CLI) bereitgestellt.¹⁷

3.1 InfluxDB Installation

Bevor InfluxDB genutzt werden kann, muss es als erstes installiert werden. Am einfachsten ist dies über Docker möglich. Dazu ist es notwendig, dass Docker und Docker Compose auf dem System installiert sind. Mit Docker Desktop lassen sich die beiden Tools am einfachsten installieren. Im Anhang dieser Arbeit befindet sich im Ordner Docker eine Docker Compose Datei mit dem Namen `docker-compose.yml`. Zum Starten der benötigten Container ist es am einfachsten mit einem Terminal (Powershell, xterm usw.) in den Docker Ordner zu wechseln und den Befehl `docker compose up -d` auszuführen. Jetzt beginnt Docker damit, die notwendigen Images herunterzuladen und zu bauen. Wenn der Befehl ohne Fehler ausgeführt worden ist, InfluxDB installiert worden und kann über die URL: `http://localhost:8086` aufgerufen werden. Die Login-Daten sind als Umgebungsvariable in Docker Compose definiert und lauten: `admin` `e1LjSYaFbzbJeIBC`.

Außerdem wurde mit diesem Befehl auch ein Jupyter Notebook in einem Docker Container gestartet. Auf diesen Container kann über die URL: `http://localhost:8888/` zugegriffen werden. Das Passwort lautet `fhdw`. Im Ordner DWD befinden sich die Notebooks mit dem in dieser Arbeit beschriebenen Code.

¹⁶vgl. solid-IT-gmbh (2022)

¹⁷vgl. Influxdata (2022a)

3.2 Daten einfügen

In InfluxDB werden Daten immer in Buckets gespeichert. Um Daten hochzuladen, muss zunächst ein Bucket angelegt werden. Dazu gibt es zwei Möglichkeiten. Die Einfachste ist es, über das Web User Interface (UI) von InfluxDB einen neuen Bucket anzulegen. Dazu muss nach dem Login der Navigationspunkt Data und der Reiter Buckets ausgewählt werden. Hier kann dann mit dem Button Create Bucket ein neuer Bucket angelegt werden. Bei dem Anlegen kann noch eine Lebensdauer für die Daten ausgewählt werden, nach welcher die jeweiligen Datenpunkte gelöscht werden.¹⁸

3.3 Line Protokoll

Daten werden immer nach dem InfluxDB Line Protokoll formatiert an die Datenbank gesendet. Das Protokoll ist wie in Listing 1 dargestellt aufgebaut. Im ersten Teil des Line Protokolls wird der Name der Messreihe angegeben. Das kann zum Beispiel der Name des Sensors sein oder der Ort an dem der Messwert genommen wurde. Wichtig ist, dass Groß und Kleinschreibung beachtet werden muss und Unterstriche nicht genutzt werden dürfen. Sonderzeichen müssen mit einem \ maskiert werden. Nach dem Namen kommen getrennt durch ein Komma die Tags der Messung. Tags werden indexiert und dazu genutzt, um Messwerte zu durchsuchen. Tags werden als Key Value Paar angegeben. Hier sollen Metadaten wie zum Beispiel der Standort des Sensors oder der Name des Servers eingetragen werden, zu dem die Datenpunkt/e gehören. Die eigentlichen Werte sind mit einem Leerzeichen von den Tags abgegrenzt und bestehen aus durch Kommas getrennten Key Value Feldern. Der letzte Wert einer Zeile ist der Unix Timestamp in Millisekunden. In einer Datei oder Anfrage kann es mehrere Zeilen mit Daten geben.¹⁹

Listing 1: InfluxDB Line Protokoll Quelle: Influxdata (2022c)

```
measurementName,tagKey=tagValue fieldKey="fieldValue" 1465839830100400200
-----|-----|-----|-----|
          |          |          |          |
      Measurement   Tag set    Field set    Timestamp
```

¹⁸vgl. Abb. 2, Abb. 3, Abb. 4, Abb. 5

¹⁹vgl. Influxdata (2022c)

Die im Line Protokoll formatierten Daten können jetzt entweder mithilfe eines Rest Requests oder des InfluxDB CLI in die Datenbank übertragen werden. Um diese Anfragen zu autorisieren muss ein API Token mitgesendet werden.²⁰ Um einen Token zu bekommen, kann dieser entweder über das Webinterface, die CLI oder über die API angelegt werden. Der einfachste Weg ist es, den Token über das Webinterface anzulegen. Dazu wird wie beim Anlegen eines Buckets zunächst der Menüpunkt Data ausgewählt und anschließend der Reiter API Tokens. Mit einem Klick auf Generate API Token kann dann ein API Token erstellt werden.²¹ Dabei kann zwischen einem All-Access token und einem Read/Write token ausgewählt werden. Mit dem All Access Token kann auf alles zugegriffen werden. Mit einem Read/Write Token kann wie in Abb. 7 zu sehen, ausgewählt werden, auf welchen Bucket geschrieben oder gelesen werden kann.²²

3.4 Daten abrufen und visualisieren

Um Daten aus InfluxDB abzurufen wird die funktionale Sprache Flux genutzt. Flux wurde für das Abfragen, Analysieren und Verarbeiten von Daten entwickelt. Flux kann unter anderem im influxDB Webinterface oder über die InfluxDB API sowie über den InfluxDB CLI Client ausgeführt werden.²³

Am einfachsten ist es Flux mit Code Beispielen zu erklären. Die Zeilenangaben in diesem Kapitel beziehen sich wenn es nicht anders angegeben ist auf das Listing 5. Damit Daten abgefragt werden können muss als erstes ein Bucket ausgewählt werden. In Zeile 1 wird dazu das from Statement genutzt um den Bucket mit den Namen test-bucket auszuwählen. Aus Performance Gründen erlaubt Flux allerdings keine Abfragen ohne die Angabe eines Zeitbereiches der die Datenmenge einschränkt. Dieser Bereich kann mithilfe des range Statements ausgewählt werden, wobei es zwei verschiedene Möglichkeiten gibt. Die erste ist mithilfe des start Parameters die Zeit relativ zum Zeitpunkt der Ausführung anzugeben. In Zeile 2 des Beispiels wird der Zeitbereich auf die letzte Stunde festgelegt. Zusätzlich zum start Parameter kann auch der Stop Parameter angegeben werden welcher wie in Zeile 5 zu sehen angibt bis wann die Daten abgerufen werden. In Zeile 5 würden die Daten von 14:00 bis 14:50 abgerufen werden wenn die

²⁰vgl. Influxdata (2022h)

²¹vgl. Abb. 2, Abb. 3, Abb. 6, Abb. 7

²²vgl. Influxdata (2022d)

²³vgl. Influxdata (2022b)

Abfrage um 15 Uhr verarbeitet wird. Die zweite Möglichkeit ist es den Zeitbereich mit absoluten Werten einzuschränken, dass wird wie in Zeile 8 zu sehen gemacht. Zusätzlich zu den Zeitbereichen können auch weitere Filter auf die Daten angewendet werden. Dazu wird die Filter Funktion genutzt. Mit dieser Funktion wird über die Datenpunkte der Datenbank iteriert. Innerhalb dieser Funktion wird eine weitere Funktion als Parameter übergeben welche die Daten filtert. Wenn die Funktion false zurückgibt wird dieser Datenpunkt verworfen, falls jedoch ein true zurückgegeben wird, werden die Daten weiter verarbeiten oder ausgegeben. Im Fall von Zeile 12 wird der Record, damit ist der aktuelle Datenpunkt in der Datenbank auf die die Funktion gerade angewendet wird, in die Variable `ř` gespeichert. Über `ř._measurement` kann nach dem Namen eines Datenpunktes und mit `ř._field` nach dem Namen eines Messwertes gefiltert werden. Im letzten Schritt werden über die `yield()` Funktion die Daten ausgegeben. Das ist nur erforderlich wenn mehrere Flux Abfragen zu einer Kombiniert werden.²⁴ Alternativ können Flux Querys auch über den Query Builder im InfluxDB Web UI erstellt werden.²⁵

Es ist nicht immer erforderlich die Daten aus der Datenbank auszulesen wenn diese Visualisiert oder verarbeiten werden sollen. Mithilfe des InfluxDB Webinterface ist es möglich Daten zu visualisieren. Bevor die Daten Visualisiert werden können müssen diese erst mithilfe eines Flux Query abgefragt werden. Dazu kann entweder selbst ein Query geschrieben werden oder der eingebaute Query Builder genutzt werden.²⁶ Danach lassen sich die Daten als Graph, Heatmap, Histogramm, Scatterplot und weiteren Diagrammtypen anzeigen.²⁷

3.5 Python library

Um nicht selbst eigene API Anfragen über die Rest API schreiben zu müssen gibt es für Python und andere Sprachen fertige Bibliotheken.²⁸ Bevor mit der Client Library gearbeitet werden kann muss diese als erste Installiert und importiert werden. Am besten wird zur Installation der Python Paketmanager pip genutzt. Mit dem Befehl `pip install influxdb-client` werden alle benötigten Packte installiert.²⁹

²⁴vgl. Influxdata (2022f)

²⁵vgl. Influxdata (2021a)

²⁶vgl. Influxdata (2021a)

²⁷vgl. Influxdata (2021b)

²⁸vgl. Influxdata (2022g)

²⁹vgl. Influxdata (2022e)

In Listing 3 kann an einem Beispiel gesehen werden wie Daten mithilfe von Python in die Datenbank geschrieben werden. Ein ähnliches Beispiel findet sich ausführbar und detailliert beschreiben im Jupyter Notebook Container in der Datei Grundlagen. In der ersten beiden Zeile des Codes wird der InfluxDB Client importiert. Danach werden in Zeile vier bis sieben die benötigten Daten bucket, Organisation, Token und Uniform Resource Locator (URL) in Variablen geschrieben. Was in die Variablen eingetragen werden muss wird im Kapitel Daten einfügen beschreiben.³⁰

In Zeile neun bis 13 wird der Client mit den hinterlegten Variablen initialisiert. Danach folgt in Zeile 15 die Initialisierung des write clients. Als nächstes muss ein Datenbank erstellt werden der in die Datenbank geschrieben werden kann. Dazu wird in Zeile 17 bis 19 der Datenpunkt erstellt. An diesen Datenpunkt kann eine beliebige Menge an Tags und Datenfeldern über die Methoden tag und field angehängt werden. Hier im Beispiel wird der Datenpunkt my_measurement mit dem Tag location angelegt, welcher den Wert Paderborn hat, und dem Datenpunkt temperature mit dem Wert 25,3 angelegt. In der letzten Zeile wird dann der write client dazu genutzt um diesen Datenbank an die Datenbank zu senden.

Um die so in die Datenbank geschriebenen Werte wieder abzurufen kann so wie in Listing 4 vorgegangen werden. Hier wird als erstes in Zeile 1 ein Query Client erstellt. Mit diesem Query Client kann dann wie in Zeile 2 bis 7 zu sehen ein FLux Query ausgeführt werden. Zurückgegeben wird ein Flux Objekt mit einer Tabellen Struktur die wie in Zeile 9 bis 11 zu sehen einfach mit einer Schleife verarbeitet werden.³¹

3.6 Wetterdaten

3.6.1 Wetterdaten Aufbau

Die Wetterdaten des DWD können über den CDC OpenData Bereich heruntergeladen werden. Hier werden die Wetterdaten über FTP und HTTPS zum Download angeboten. Unter der URL https://www.dwd.de/DE/leistungen/cdc/cdc_ueberblick-klimadaten.html wird eine gute Übersicht über die zum Download angeboten Daten geboten. Die Werte für die aktuelle Lufttemperatur können über https://opendata.dwd.de/climate_environment/CDC/observations_germany/climate/10_minutes/air_temperature/

³⁰vgl. Influxdata (2022e)

³¹vgl. Influxdata (2022e)

now/ abgerufen werden. Historische Daten können über https://opendata.dwd.de/climate_environment/CDC/observations_germany/climate/10_minutes/air_temperature/now/ abgerufen werden.

Aktuell werden auf der Webseite, für die aktuelle Lufttemperatur, ca 480 Dateien zum Download angeboten. Die meisten dieser Dateien entsprechen jeweils einer Messstation und je nach Tageszeit kann deswegen die Menge der Zeilen in der Datei variieren, weil immer um 00:00 eine neue Datei angefangen wird. In den Zip-Dateien finden sich außerdem Metadaten über die Messstationen. Die eigentlichen Daten sind als Comma-separated values (CSV) formatiert und sehen aus wie in Listing 2 gekürzt dargestellt. In der CSV Datei gibt es 9 Felder. Der Inhalt der Felder wird in Tabelle 1 beschreiben.

Listing 2: Wetterdaten CSV Quelle: DWD Wetterdaten CSV Download

```
STATIONS_ID;MESS_DATUM; QN;PP_10;TT_10;TM5_10;RF_10;TD_10;eor
73;202205120000; 2; -999; 12.9; 11.2; 84.2; 10.3;eor
73;202205120010; 2; -999; 12.7; 11.2; 84.9; 10.2;eor
73;202205120020; 2; -999; 12.9; 11.4; 83.0; 10.1;eor
73;202205120030; 2; -999; 12.4; 10.7; 86.9; 10.3;eor
73;202205120040; 2; -999; 12.4; 10.5; 86.2; 10.2;eor
73;202205120050; 2; -999; 12.3; 10.3; 85.5; 9.9;eor
73;202205120100; 2; -999; 12.1; 10.1; 88.1; 10.2;eor
73;202205120110; 2; -999; 11.7; 9.9; 90.1; 10.1;eor
73;202205120120; 2; -999; 11.7; 10.0; 89.0; 10.0;eor
```

In der Datei `zehn_now_tu_Beschreibung_Stationen.txt` werden die Wetterstationen beschrieben. Diese Datei ist nicht als CSV Datei formatiert sondern als Tabelle und erhält Daten über die Wetterstationen. Die Daten der Stationen in der heruntergeladenen Textdatei stimmen mit den Daten der Hauptamtliches Messnetz Karte überein. Allerdings enthält die Textdatei nicht alle Stationen sondern nur Station für die auch Messwerte im Datensatz hinterlegt sind. Die Bedeutung der einzelnen Spalten der Tabelle sind in der Tabelle 2 beschreiben.

Tabelle 1: Bedeutung der CSV Felder

Feld Name	Bedeutung
STATION_ID	Gibt an von welcher Station die Werte stammen
MESS_DATUM	Gibt an wann gemessen wurde im Format %Y%m%d%H%M. Also Jahr Monat Tag Stunde Minute als eine zusammengeschriebene Zahl.
QN	Gibt die Qualität der Messwerte an. Hier gibt es die Werte 1 bis 3 1. nur formale Kontrolle bei Dekodierung und Import 2. Kontrolle mit individuell festgelegten Kriterien 3. ROUTINE automatische Kontrolle und Korrektur mit Software (QUALIMET)
PP_10	Luftdruck auf Stationshöhe
TT_10	Lufttemperatur auf 2m Höhe
TM5_10	Lufttemperatur auf 5cm Höhe
TD_10	relative Luftfeuchtigkeit auf 2m Höhe
eor	END OF RECORD"Bedeutet die Zeile ist zu Ende.

Quelle: Eigene Darstellung https://opendata.dwd.de/climate_environment/CDC/observations_germany/climate/10_minutes/air_temperature/now/BESCHREIBUNG_obsgermany_climate_10min_tu_now_de.pdf

Tabelle 2: Bedeutung Stationstabellen Felder

Feld Name	Bedeutung
STATION_ID	Gibt an von welcher Station die Werte stammen
von_datum	Datum seit dem die Station aktiv ist.
bis_datum	Hohe der Station.
Stationshoehe	Hohe über dem Normalnullpunkt
geoBreite	Breitengrad der Station.
geoLaenge_10	Längengrad der Stations.
Stationsname	Name der Station.
Bundesland	Bundesland in dem die Station steht.

Quelle: Vergleich der Werte mit der Hauptamtliches Messnetz Karte https://opendata.dwd.de/climate_environment/CDC/observations_germany/climate/10_minutes/air_temperature/now/zehn_now_tu_Beschreibung_Stationen.txt

3.6.2 Wetterdaten abrufen

Um die Daten auswerten zu können müssen diese als erstes heruntergeladen und entpackt werden. Dazu wird mithilfe von BeautifulSoup aus der Hypertext Markup Language (HTML) Seite des DWD für jede Datei eine URL ausgelesen. Die so gewonnen URLs können dann mithilfe einer Schleife heruntergeladen werden. Um den Messwerten eine Station zuzuordnen zu können wird als erstes die Datei mit den Stationen verarbeitet. Für jede Station wird ein Objekt erstellt und in ein Dictionary gespeichert. Dadurch kann in dem Dictionary einfach über die STATIONS_ID die passende Station gefunden werden. Weil diese Datei allerdings nicht CSV formatiert ist, muss die Datei auf eine andere Art ausgewertet werden. Um die einzelnen Felder aus einer Zeile zu bekommen, wird immer so lange gelesen, bis wieder eine bestimmte Anzahl von Leerzeichen hintereinander erkannt worden ist. Die Zeichen zwischen den Leerzeichen sind dann ein ausgelesenes Feld. Nachdem die Stationsdaten ausgewertet worden sind, werden die CSV Dateien in einer Schleife entpackt und mithilfe der Bibliothek Pandas in ein Dataframe umgewandelt. Das so erzeugte Dataframe wird im letzten Schritt mit den Daten der Stationsdaten zusammengeführt und als Datenpunkt in InfluxDB geschrieben. Weitere Erklärungen und der Code selbst kann im angehängten Jupyter Notebook eingesehen und ausgeführt werden.

3.6.3 Wetterdaten Verarbeiten

4 Kritische Auseinandersetzung

An dieser Arbeit kann kritisiert werden, dass viele der genutzten Quellen von InfluxDB veröffentlicht wurden. Zusätzlich wurden sehr viele Internetquellen zitiert und nur wenige Monographien. Außerdem konnten durch die begrenzte Zeit und den begrenzten Umfang dieser Arbeit nicht alle Funktionen und Besonderheiten von TSDB und InfluxDB im Besonderen beschrieben werden. Weitere Informationen dazu können in der InfluxDB Dokumentation und den genutzten Quellen gefunden werden.

5 Zusammenfassung

Time Series Data ist eine Reihe von Daten, die die Veränderung eines Zustandes über eine Zeitspanne aufzeichnen. Diese Daten können IOT Sensor Daten, Servermetriken, Netzwerkdaten, Marktgeschäfte oder viele andere Arten von Daten sein.³² Time Series Datenbanken sind genau dafür entwickelt worden, um effizient mit dieser Art von Daten zu arbeiten und sind dabei um einiges schneller als RDBMS. Allerdings sind sie auch weniger flexibel und haben kein so ausgeprägtes Ökosystem.³³

Mithilfe von Docker und Jupyter Notebook kann sehr schnell eine Entwicklungsumgebung aufgebaut werden, mit der einfach die als Beispiel genutzten Wetterdaten heruntergeladen, importiert und verarbeitet werden können.

6 Schlussfolgerung und Fazit

TSDB eignen sich sehr gut für die Speicherung von Time Series Data und besonders für die Verarbeitung von großen Datenmengen. Dadurch, dass immer mehr Daten erfasst werden, ist es abzusehen, dass sich TSDB immer weiter verbreiten werden. Es bietet sich an, für IOT Sensoren oder die Erfassung von Servermetriken und anderer Time Series Data auf TSDB zu setzen.

³²vgl. ComputerWeekly.de, Redaktion (2021)

³³vgl. Influxdata (2021c)

Anhang

Anhangsverzeichnis

Anhang 1:	InfluxDB Python Code	17
Anhang 2:	Times Seires DB Rangliste	18
Anhang 3:	InfluxDB Flux Code	19
Anhang 4:	InfluxDB Webinterface Screenshot	20

Anhang 1 InfluxDB Python Code

Listing 3: InfluxDB Python Daten schreiben Quelle: Influxdata (2022e)

```
1  import influxdb_client
2  from influxdb_client.client.write_api import SYNCHRONOUS
3
4  bucket = "<my-bucket>"
5  org = "<my-org>"
6  token = "<my-token>"
7  url="http://influxdb:8086"
8
9  client = influxdb_client.InfluxDBClient(
10     url=url,
11     token=token,
12     org=org
13 )
14
15 write_api = client.write_api(write_options=SYNCHRONOUS)
16
17 p = influxdb_client.Point("my_measurement")
18     .tag("location", "Paderborn")
19     .field("temperature", 25.3)
20 write_api.write(bucket=bucket, org=org, record=p)
```

Listing 4: InfluxDB Python Daten lesen Quelle: Influxdata (2022e)

```
1  query_api = client.query_api()
2  query = 'from(bucket:"my-bucket")\
3  |> range(start: -10m)\
4  |> filter(fn:(r) => r._measurement == "my_measurement")\
5  |> filter(fn: (r) => r.location == "Prague")\
6  |> filter(fn:(r) => r._field == "temperature" )'
7  result = query_api.query(org=org, query=query)
8  results = []
9  for table in result:
10     for record in table.records:
11         results.append((record.get_field(), record.get_value()))
12
13 print(results)
14 [(temperature, 25.3)]
```


Anhang 2 Times Seires DB Rangliste

Abbildung 1: DB-Engines Ranking

394 Systeme im Ranking, Mai 2022

Rang			DBMS	Datenbankmodell	Punkte		
Mai 2022	Apr 2022	Mai 2021			Mai 2022	Apr 2022	Mai 2021
1.	1.	1.	Oracle +	Relational, Multi-Model	1262,82	+8,00	-7,12
2.	2.	2.	MySQL +	Relational, Multi-Model	1202,10	-2,06	-34,28
3.	3.	3.	Microsoft SQL Server +	Relational, Multi-Model	941,20	+2,74	-51,46
4.	4.	4.	PostgreSQL +	Relational, Multi-Model	615,29	+0,83	+56,04
5.	5.	5.	MongoDB +	Document, Multi-Model	478,24	-5,14	-2,78
6.	6.	↑ 7.	Redis +	Key-value, Multi-Model	179,02	+1,41	+16,85
7.	↑ 8.	↓ 6.	IBM Db2	Relational, Multi-Model	160,32	-0,13	-6,34
8.	↓ 7.	8.	Elasticsearch +	Suchmaschine, Multi-Model	157,69	-3,14	+2,34
9.	9.	↑ 10.	Microsoft Access	Relational	143,44	+0,66	+28,04
10.	10.	↓ 9.	SQLite +	Relational	134,73	+1,94	+8,04
11.	11.	11.	Cassandra +	Wide column	118,01	-3,98	+7,08
12.	12.	12.	MariaDB +	Relational, Multi-Model	111,13	+0,81	+14,44
13.	13.	13.	Splunk	Suchmaschine	96,35	+1,11	+4,24
14.	14.	↑ 27.	Snowflake +	Relational	93,51	+4,06	+63,46
15.	15.	15.	Microsoft Azure SQL Database	Relational, Multi-Model	85,33	-0,45	+14,88
16.	16.	16.	Amazon DynamoDB +	Multi-Model	84,46	+1,55	+14,39
17.	17.	↓ 14.	Hive +	Relational	81,61	+0,18	+5,42
18.	18.	↓ 17.	Teradata +	Relational, Multi-Model	68,39	+0,82	-1,59
19.	19.	19.	Neo4j +	Graph	60,14	+0,62	+7,91
20.	20.	20.	Solr	Suchmaschine, Multi-Model	57,26	-0,48	+6,07
21.	21.	↓ 18.	SAP HANA +	Relational, Multi-Model	55,09	-0,71	+2,33
22.	22.	22.	FileMaker	Relational	52,27	-0,64	+5,55
23.	↑ 24.	↑ 24.	Google BigQuery +	Relational	48,61	+0,63	+10,98
24.			Databricks	Multi-Model	47,85		
25.	↓ 23.	↓ 21.	SAP Adaptive Server	Relational, Multi-Model	47,78	-0,58	-2,19
26.	↓ 25.	↓ 23.	HBase +	Wide column	43,19	-1,14	-0,05
27.	↓ 26.	↓ 25.	Microsoft Azure Cosmos DB +	Multi-Model	40,22	-0,12	+5,51
28.	↓ 27.	28.	PostGIS	Spatial DBMS, Multi-Model	31,82	-0,23	+1,98
29.	↓ 28.	29.	InfluxDB +	Time Series, Multi-Model	29,55	-0,47	+2,38
30.	↓ 29.	↓ 26.	Couchbase +	Document, Multi-Model	28,38	-0,67	-1,85

Quelle: <https://db-engines.com/de/ranking?msclkid=4f2a29e5d08811ec95ccd74f8f5146ab>

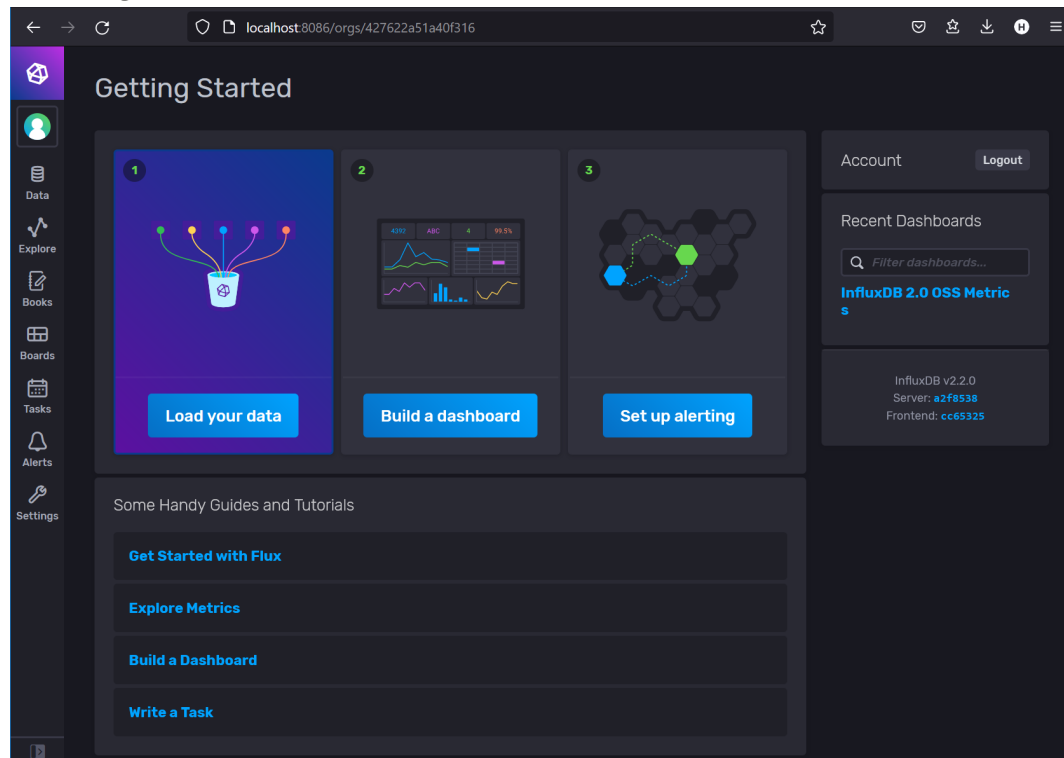
Anhang 3 InfluxDB Flux Code

Listing 5: InfluxDB Flux Beispiel Quelle: Influxdata (2022e)

```
1   from(bucket:"test-bucked")
2     |> range(start: -1h)
3
4   from(bucket:"test-bucked")
5     |> range(start: -1h, stop: -10m)
6
7   from(bucket:"test-bucked")
8     |> range(start: 2022-06-30T00:00:00Z, stop: 2022-06-30T12:00:00Z)
9
10  from(bucket: "test-bucked")
11    |> range(start: -15m)
12    |> filter(fn: (r) => r._measurement == "cpu" and r._field == "cpu-total")
13    |> yield()
```

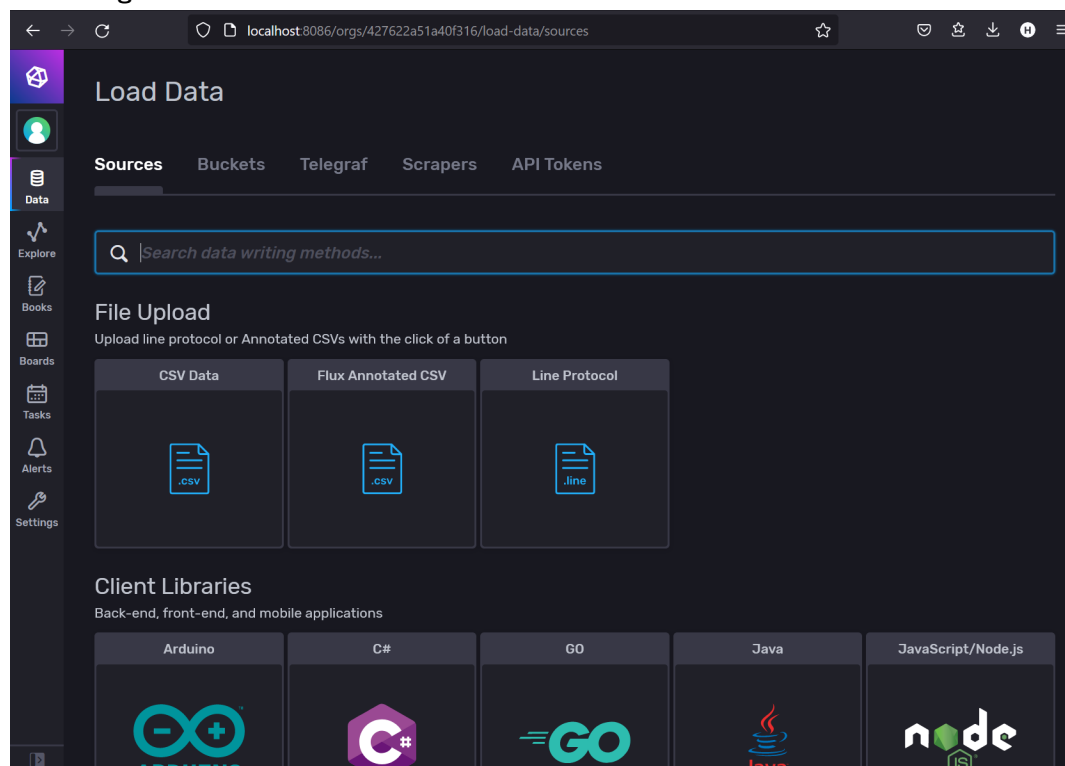
Anhang 4 InfluxDB Webinterface Screenshot

Abbildung 2: InfluxDB Dashboard



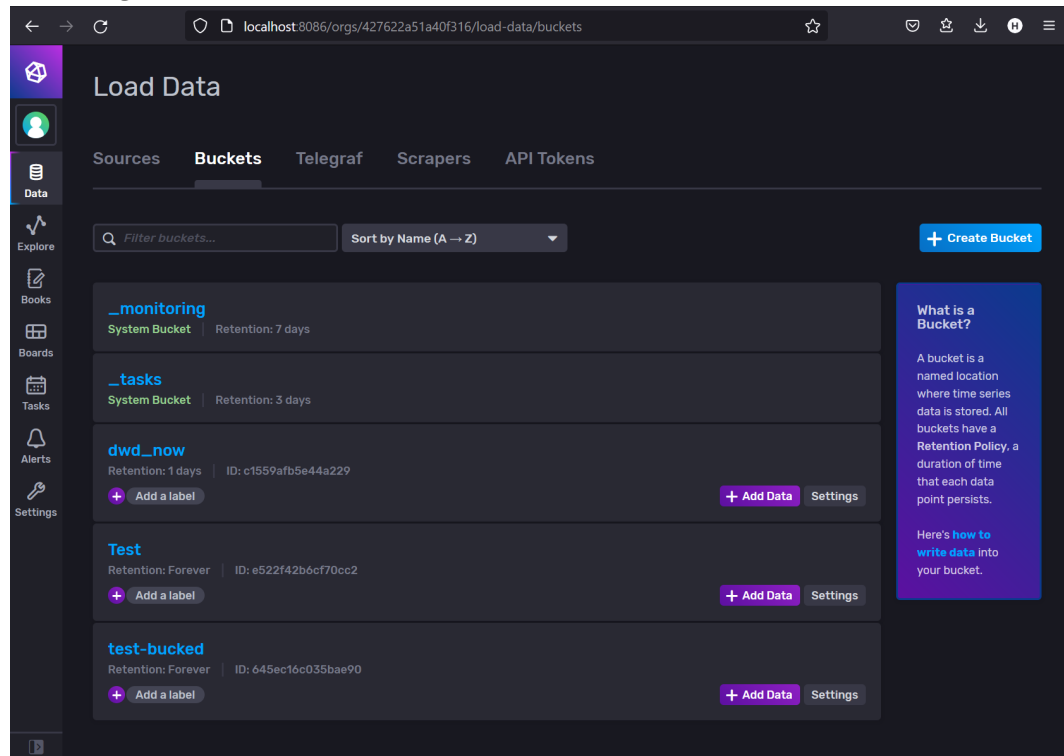
Quelle: Eigener Screenshot InfluxDB Webinterface

Abbildung 3: InfluxDB Load Data Source



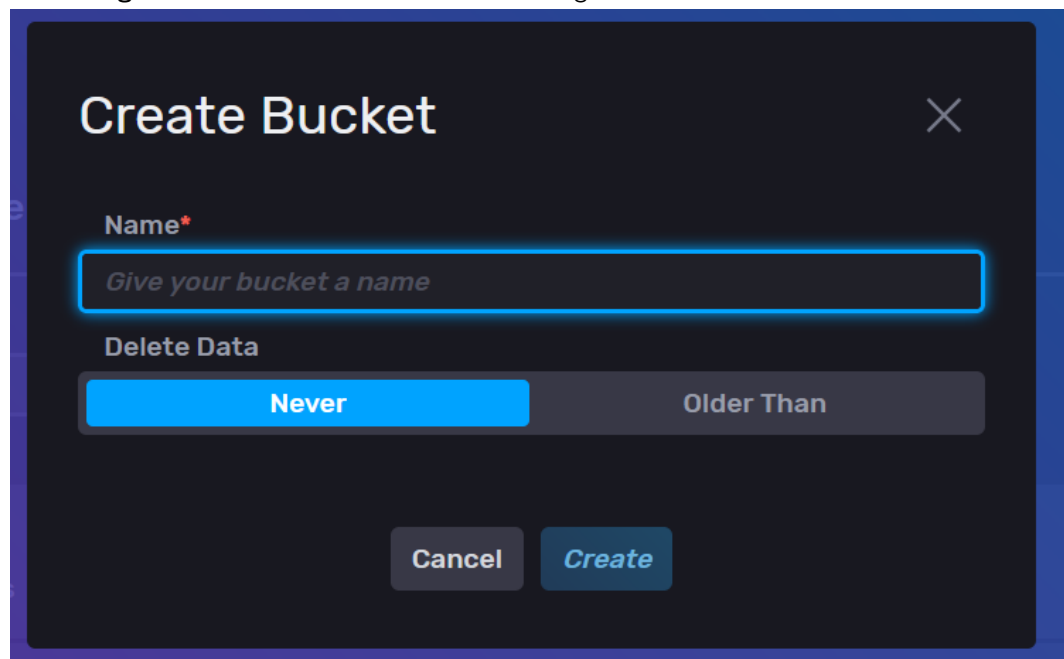
Quelle: Eigener Screenshot InfluxDB Webinterface

Abbildung 4: InfluxDB Load Data Bucket

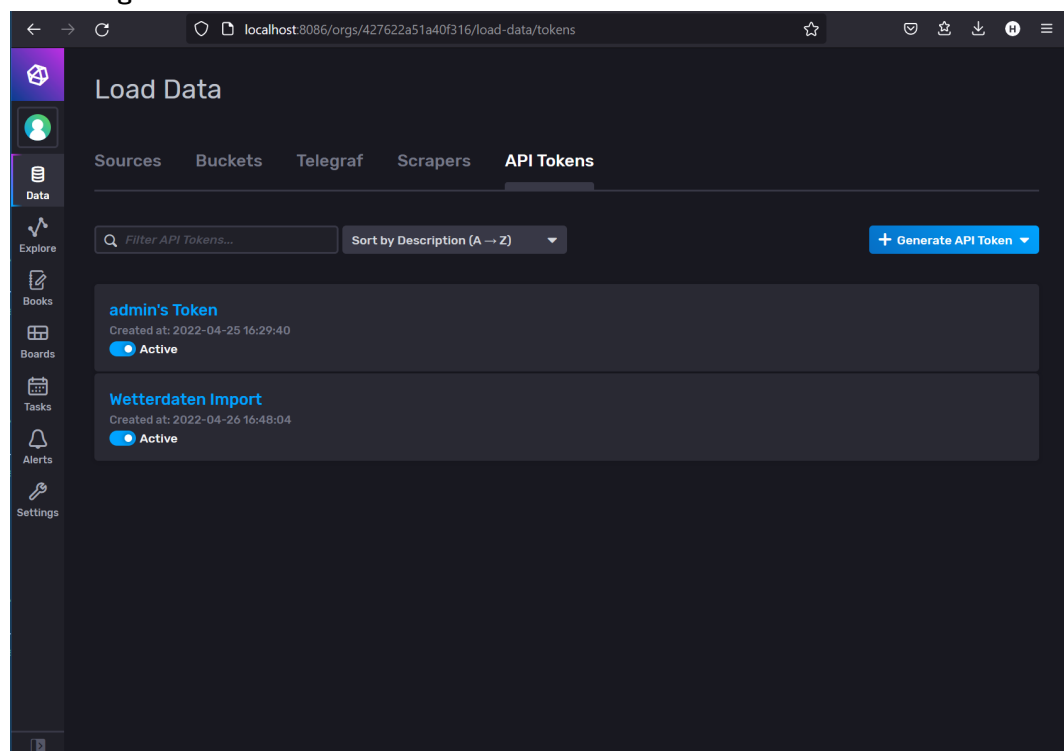


Quelle: Eigener Screenshot InfluxDB Webinterface

Abbildung 5: InfluxDB Load Data Bucket hinzufügen



Quelle: Eigener Screenshot InfluxDB Webinterface

Abbildung 6: InfluxDB Load Data API Tokens

Quelle: Eigener Screenshot InfluxDB Webinterface

Abbildung 7: InfluxDB Load Data API Token hinzufügen

Generate Read/Write API Token

Description

Describe this new API token

Read

All Buckets **Scoped**

BUCKETS Select All Deselect All

Search buckets...

Test
dwd_now
test-bucked

Write

All Buckets **Scoped**

BUCKETS Select All Deselect All

Search buckets...

Test
dwd_now
test-bucked

Cancel Save

Quelle: Eigener Screenshot InfluxDB Webinterface

Quellenverzeichnis

Monographien

Silaparasetty, Nikita (2020). *Machine Learning Concepts with Python and the Jupyter Notebook Environment*. Apress Berkeley CA, S. 91–118.

Stender, Daniel (2020). *Cloud-Infrastrukturen: Infrastructure as a Service – So geht moderne IT-Infrastruktur. Das Handbuch für DevOps-Teams und Administratoren*. Rheinwerk Computing, S. 54–68.

Internetquellen

ComputerWeekly.de, Redaktion (2021). *Definition Zeitreihendatenbank (Time Series Database, TSDB)*. URL: <https://datascience.eu/wiki/what-the-heck-is-time-series-data-and-why-do-i-need-a-time-series-database/> (besucht am 21. Mai 2021).

Data-Science-Team (2020). *What the heck is time-series data (and why do I need a time-series database)?* URL: <https://www.computerweekly.com/de/definition/Zeitreihendatenbank-Time-Series-Database-TSDB> (besucht am 9. Mai 2022).

Dix, Paul (2021). *Why Time Series Matters for Metrics, Real-Time Analytics and Sensor Data*. URL: <http://get.influxdata.com/rs/972-GDU-533/images/why%20time%20series.pdf> (besucht am 10. Mai 2022).

Fangman, Sam (2019). *The Time Has Come for a New Type of Database*. URL: <https://medium.datadriveninvestor.com/the-time-has-come-for-a-new-type-of-database-47cf8df1667a> (besucht am 10. Mai 2022).

Hajek Vlasta Pour Ales, Kudibal Ivan (2019). *Why Time Series Matters for Metrics, Real-Time Analytics and Sensor Data*. URL: <http://get.influxdata.com/rs/972-GDU-533/images/why%20time%20series.pdf> (besucht am 27. Mai 2022).

- hazelcast (2022). *Time Series Database*. URL: <https://hazelcast.com/glossary/time-series-database/> (besucht am 10. Mai 2022).
- Influxdata (2021a). *Explore metrics with InfluxDB*. URL: <https://docs.influxdata.com/influxdb/cloud/visualize-data/explore-metrics/> (besucht am 20. Mai 2022).
- Influxdata (2021b). *Visualization types*. URL: <https://docs.influxdata.com/influxdb/cloud/visualize-data/visualization-types/> (besucht am 20. Mai 2022).
- Influxdata (2021c). *Why You Should Migrate from SQL to NoSQL for Time Series Data*. URL: <https://www.influxdata.com/from-sql-to-nosql/> (besucht am 20. Mai 2022).
- Influxdata (2022a). *API Quick Start*. URL: https://docs.influxdata.com/influxdb/v2.2/api-guide/api_intro/ (besucht am 21. Mai 2022).
- Influxdata (2022b). *Execute queries*. URL: <https://docs.influxdata.com/influxdb/cloud/query-data/execute-queries/> (besucht am 21. Juni 2022).
- Influxdata (2022c). *Line protocol*. URL: <https://docs.influxdata.com/influxdb/v2.2/reference/syntax/line-protocol/> (besucht am 29. Mai 2022).
- Influxdata (2022d). *Manage API tokens*. URL: <https://docs.influxdata.com/influxdb/cloud/security/tokens/#all-access-token> (besucht am 29. Mai 2022).
- Influxdata (2022e). *Python client library*. URL: <https://docs.influxdata.com/influxdb/v2.2/api-guide/client-libraries/python/> (besucht am 18. Juni 2022).
- Influxdata (2022f). *Query InfluxDB with Flux*. URL: <https://docs.influxdata.com/influxdb/cloud/query-data/get-started/query-influxdb/> (besucht am 21. Mai 2022).

Influxdata (2022g). *Use InfluxDB client libraries*. URL: <https://docs.influxdata.com/influxdb/v2.2/api-guide/client-libraries/> (besucht am 18. Juni 2022).

Influxdata (2022h). *Write data with the InfluxDB API*. URL: <https://docs.influxdata.com/influxdb/v2.2/write-data/developer-tools/api/> (besucht am 29. Mai 2022).

Seeger, Marc (2009). *Key-Value stores: a practical overview*. URL: http://blog.marc-seeger.de/assets/papers/Ultra_Large_Sites_SS09-Seeger_Key_Value_Stores.pdf (besucht am 19. Juni 2022).

solid-IT-gmbh (2022). *DB-Engines Ranking*. URL: <https://db-engines.com/de/ranking> (besucht am 10. Mai 2022).

Ehrenwörtliche Erklärung

Hiermit erkläre ich, dass ich die vorliegende Studienarbeit - Entwurf selbständig angefertigt habe. Es wurden nur die in der Arbeit ausdrücklich benannten Quellen und Hilfsmittel benutzt. Wörtlich oder sinngemäß übernommenes Gedankengut habe ich als solches kenntlich gemacht. Diese Arbeit hat in gleicher oder ähnlicher Form noch keiner Prüfungsbehörde vorgelegen.

Paderborn, 24. Juni 2022

Henrik Mertens