Bild: Albert Hulm

Geschichtsschreiber

InfluxDB: Spezialisierte Datenbank für Messwerte und Logging

Eine SQL-Datenbank ist meist die erste Wahl für alle Arten von Daten. Für Messwerte ist der Generalist aber nicht unbedingt die beste Lösung. Eine Zeitreihendatenbank wie InfluxDB arbeitet schneller und kann alte Daten automatisch zusammenfassen.

Von Jan Mahn

Messwerte können aus vielen Quellen stammen: zum Beispiel von physischen Sensoren, seien sie im vernetzten Zuhause, im Rechenzentrum oder in einer Industrieanlage. Aber auch Software kann Messwerte produzieren: Prozessor- und Netzwerkauslastung, Anzahl von Anfragen oder Bestellungen. Gründe, solche Messwerte zu speichern, gibt es ebenfalls viele: um Ursachen für Probleme zu finden, Durchschnittswerte zu berechnen oder übersichtliche Diagramme zu generieren.

Wer schon eine relationale Datenbank wie MySQL, MariaDB oder SQL Server im Einsatz hat, ist versucht, diese Datenbank auch für Zeitreihendaten, also die Zuordnung von Messwerten zu Datum und Uhrzeit, einzusetzen. Um zu verstehen, welches Problem Zeitreihendatenbanken lösen, kann man das Vorgehen bei einer MySQL-Datenbank einmal durchdenken. Es genügt eine Tabelle mit drei Spalten: Name des Messwerts, gemessener Wert, Zeitstempel. Das Speichern klappt mit `INSERT INTO` noch problemlos.

Datenbergbau

Probleme gibt es aber bei der Auswertung großer Datenmengen. Angenommen, 10 Temperatursensoren melden alle 30 Sekunden einen Messwert an die Datenbank. Das ergibt 28.800 Messwerte am Tag und über 10 Millionen im Jahr. Bei der Auswertung stellt man aber schnell fest, dass die einzelnen Messwerte für sich wertlos sind – interessant ist nicht die Information, dass die Temperatur am 5. Juni 2018 um 12:34:56 genau 25,1 Grad betrug. Stattdessen sucht man nach aggregierten Daten, also zusammenfassenden Werten für bestimmte Zeitabschnitte, zum Beispiel Höchst- und Niedrigstwerte pro

Tag oder Durchschnittswerte pro Stunde. Solche Berechnungen kann man durchaus mit SQL-Abfragen immer dann generieren, wenn sie jemand lesen will. Dazu bastelt man ein Skript, das geschickt mit `GROUP BY`-Anweisungen und den Datumsfunktionen der verwendeten Datenbank hantiert und Daten aus der MySQL-Datenbank holt. Wer eine solche Lösung im Einsatz und viele Messwerte in der Datenbank hat, bemerkt aber irgendwann, dass das Aggregieren spürbar Rechenzeit kostet. Soll aus den Durchschnittswerten ein Diagramm gezeichnet werden, nervt die Wartezeit richtig.

Schöner wäre es doch, die Daten regelmäßig zu Blöcken zusammenzufassen, die Aggregate in einer neuen Tabelle abzuspeichern und sich von alten Daten zu trennen. Kein Problem für den SQL-Entwickler – mit einem weiteren Skript und einem Cron-Job unter Linux oder einer geplanten Aufgabe unter Windows. Wirklich übersichtlich und leicht zu warten ist diese Lösung aber nicht. Perfekt wäre es dagegen, wenn die Datenbank diese Aufgaben selbst übernehmen könnte. All diese Funktionen bekommt man, wenn man seine Daten gleich in einer Zeitreihendatenbank ablegt, die genau für diese Anwendung erfunden wurde.

Inselbegabung

Große Verbreitung unter den Zeitreihendatenbanken (Time Series Databases) hat InfluxDB gefunden. Die Software gibt es in einer Open-Source-Ausgabe, die fast alle Funktionen enthält. Nur für den Aufbau eines Clusters aus mehreren Datenbanken muss man zahlen. InfluxDB gehört zu den NoSQL-Datenbanken – das Schema mit den Tabellenspalten muss also nicht definiert werden, bevor man Werte speichern kann. Es entsteht beim Einfügen von Daten automatisch.

Eine Testinstanz von InfluxDB richten Sie auf Ihrem Linux-, Windows- oder macOS-System am schnellsten mit Docker ein. Wie Sie das installieren und wie Container funktionieren, haben wir in einem Online-Artikel auf dem aktuellen Stand zusammengefasst, zu finden über ct.de/yyyw. Für InfluxDB gibt es einen offiziellen Docker-Container. Für den Einstieg reicht der folgende Befehl:

```
docker run --name=influxdb↵
↳ -d -p 8086:8086 influxdb
```

Docker startet InfluxDB und verbindet Port 8086 der Netzwerkschnittstelle mit Port 8086 im Container. Diesen nutzt die Datenbank für die Kommunikation. Mit

```
docker exec -it influxdb influx
```

landen Sie im InfluxDB-Kommandozeilenprogramm im Container und können direkt Befehle absetzen. Zunächst braucht das Beispielprojekt eine Datenbank:

```
CREATE DATABASE measure
```

Mit `USE measure` wird diese ab jetzt verwendet. Eine Datenbank ist die Hülle für mehrere Messwertreihen. Die ersten Messpunkte einer Beispiel-Temperaturmessung sollen zur Messreihe „temp“ gehören und zwei Werte enthalten, einen für „inside“ und einen für „outside“:

```
INSERT temp inside=24.2,outside=10.1
```

InfluxDB entscheidet selbst, welcher Datentyp sinnvoll ist – in diesem Fall zwei Floats. Auch Strings, Integer und Bool-Werte sind möglich. Versucht man später, einen Wert eines anderen Typs zu speichern, beschwert sich der Server. Kein Problem ist es aber, später eine Zeile mit einem dritten Wert, zum Beispiel einen String mit dem Namen „middle“ abzuspeichern – einer der Vorteile einer NoSQL-Datenbank.

Jeder Messpunkt bekommt einen Zeitstempel. Gibt man keinen an, setzt InfluxDB den aktuellen. Das sieht man beim Auslesen der gespeicherten Werte. SQL-Nutzern kommt die Syntax bekannt vor:

```
SELECT * FROM "temp"
```

InfluxDB arbeitet mit Unix-Timestamps in Nanosekunden – die Zahl gibt die Nanosekunden seit dem 1. Januar 1970 an. Wer das nicht im Kopf umrechnen möchte, kann das Kommandozeilenprogramm dazu bringen, ein lesbares Format auszugeben:

```
precision rfc3339
```

Auch die Formatierung der Ergebnisse einer Suche kann man ändern. Möchte man die Werte nicht als Tabelle auf der Kommandozeile betrachten, reichen die Befehle `format json` oder `format csv` für eine JSON- oder CSV-Darstellung.

Hat man mehrere Sensoren, muss man nicht für jeden Sensor eine Messreihe eröffnen (mit `INSERT temp_room2`). Stattdessen sollte man mit Tags arbeiten, die man an Messwerte anheftet. Diese werden im `INSERT`-Befehl direkt nach dem Namen der Messreihe, getrennt durch ein Komma, angegeben. Die Sensorwerte im Beispiel sollen mit einem Namen des Raums und des Gebäudes versehen werden:

```
INSERT temp,room=bad,↵
↳building=ferienhaus ↵
↳inside=22.9,outside=11.5
```

Nach den Tags folgt ein Leerzeichen, alles nach diesem interpretiert InfluxDB als Messwert. Nach Tags kann man später effizient filtern. Sie werden von InfluxDB automatisch indiziert, sind also schnell durchsuchbar. Auch die Filter-Syntax entspricht der aus der SQL-Welt:

```
SELECT * FROM temp WHERE↵
↳ "room" = 'bad'
```

Auch nach Messwerten kann man mit `WHERE` filtern. Das dauert aber, weil die Werte nicht indiziert sind, etwas länger als die Suche nach Tags.

Möchte man die Daten aus InfluxDB ansprechend visualisieren, kann man auf das Open-Source-Programm Grafana setzen.

```
CREATE CONTINUOUS QUERY "temp_1h" ON "temp" BEGIN
  SELECT mean("inside") AS "mean_inside",mean("outside") AS "mean_outside",
  max("inside") AS "max_inside, max("outside") AS "max_outside
  INTO "oneyear"."temp_calc"
  FROM "temp"
  GROUP BY time(60m)
END
```

Diese Abfrage fasst alle 60 Minuten die Messwerte zu Mittel- und Maximalwerten zusammen und speichert diese in „temp_calc“.

Wegwerfen

Nachdem die Werte in der Datenbank landen, ist es an der Zeit, die automatische Zusammenfassung und Löschung einzurichten. Dazu braucht es eine „Retention Policy“, also eine Regel zur Aufbewahrung von Daten. Die bekommt einen sprechenden Namen, den Namen der Datenbank, in der sie angelegt wird, und eine Laufzeit. Außerdem muss man angeben, auf wie vielen Instanzen die Werte vorgehalten werden – in der Open-Source-Version immer nur einmal. Angelegt wird die Richtlinie mit einem `CREATE`-Befehl:

```
CREATE RETENTION POLICY ↵
↳"oneyear" ON "measure" ↵
↳DURATION 52w REPLICATION 1
```

Die Regel mit dem Namen „oneyear“ existiert jetzt, hält die Daten 52 Wochen vor, wird aber noch nicht angewendet. Um Messwerte zu speichern, auf die die Regel angewendet wird, übergibt man ihren Namen im `INSERT INTO`-Befehl:

```
INSERT INTO oneyear temp,room=bad,↵
↳building=ferienhaus ↵
↳inside=20.1,outside=9.4
```

Soll eine Regel immer dann gelten, wenn man keine andere Regel angibt, kann man sie beim Anlegen mit `DEFAULT` am Ende als Standardregel definieren.

Zusammenfassen

Die Lösung von InfluxDB für automatische Zusammenfassungen von Daten heißt „Continuous Query“. Als Anwender muss man eine solche nur anlegen, um die Ausführung kümmert sich InfluxDB. Die Beispiel-Temperaturwerte sollen zu einstündigen Blöcken zusammengefasst werden – jeweils ein Durchschnittswert (englisch „mean“) und der Maximalwert für innen und außen sollen in einer neuen Messwertreihe „temp_calc“ landen und dort ein Jahr lang gespeichert bleiben. Den vollständigen Befehl finden Sie auf dieser Seite.

Innerhalb des Befehls muss zwingend eine `GROUP BY`-Anweisung vorkommen, die angibt, wie groß die Zeitabschnitte sein sollen. Neben Funktionen wie `mean()` und `max()` versteht InfluxDB noch zahlreiche weitere Rechenoperationen. Die Dokumentation, zu finden über ct.de/yyyw, listet die Möglichkeiten auf. Mit einem `SELECT`-Aufruf kann man sich das Ergebnis ansehen – vorausgesetzt, man wartet eine Stunde oder gruppiert in einem kürzeren Zeitraum. Die Aggregate zeigt:

```
SELECT * FROM temp_calc
```

In der Praxis

Um eine InfluxDB-Instanz in der Praxis einzusetzen, sollte man sich erst einmal einen Plan machen, welche Daten wie lange als Rohdaten vorliegen sollen und welche Aggregate man später braucht – diese Denkarbeit kann die Datenbank nicht abnehmen. Die Genauigkeit kann mit zunehmendem Alter abnehmen: Zwei Wochen lang könnten alle Messwerte für die schnelle Fehlerdiagnose bleiben, zwei Monate lang die stündlichen Zusammenfassungen, dann zehn Monate lang Tageshöchstwerte und so weiter.

Verarbeitet man personenbezogene Daten in der Datenbank, wird der Datenschutzbeauftragte ein Wort bei der Entscheidung über die Zeiträume mitreden. Mit `SHOW RETENTION POLICIES` bekommt man eine Übersicht über alle angelegten Richtlinien und kann ihm so schnell belegen, wie lange Daten gespeichert werden.

Anders als in der Testumgebung möchte man später nicht über die Kommandozeile mit der Datenbank sprechen. Ein großer Vorteil gegenüber vielen SQL-Datenbanken: InfluxDB nutzt kein eigenes Netzwerkprotokoll, für das man einen eigenen Client bräuchte. Stattdessen stellt der Server ein REST-API über HTTP bereit. Die Anwendung muss also nur HTTP sprechen können. Der Endpunkt zum Schreiben von Werten heißt `/write`. Wer die Kommunikation über das Netzwerk testen will, kann ein weiteres Kommandozeilenfenster öffnen und mit Curl einen Messwert schreiben:

```
curl -i -XPOST "http://localhost:8086↵
↳/write?db=measure" ↵
↳--data-binary 'temp,room=bad ↵
↳inside=15.6'
```

Für viele gängige Programmiersprachen gibt es aber Bibliotheken, die die passenden HTTP-Aufrufe erstellen, sodass man umhinkommt, die komplette Dokumentation für das API zu studieren. Möchte man InfluxDB einsetzen, um Messwerte aus der Hausautomation zu speichern, muss man das Rad ebenfalls nicht neu erfinden. So gibt es etwa für Node-Red ein Plug-in, mit dem man die Daten ganz ohne Programmierarbeit in die Datenbank bekommt.

Wenn Sie sich dafür entscheiden, InfluxDB nicht nur zu Testzwecken im Docker-Container einzusetzen, müssen Sie dem Container ein Docker-Volume übergeben, damit die Daten auch nach einem Neustart des Containers erhalten bleiben. InfluxDB erwartet die Daten innerhalb des Containers im Ordner „/var/lib/influxdb“. Außerdem sollten Sie sich Gedanken darüber machen, welche Applikation auf den Server zugreifen kann – in der Standardkonfiguration steht das HTTP-API ohne Anmeldung offen. Die Dokumentation erklärt, wie Sie Benutzer für Datenbanken einrichten und Zugriffsrechte anlegen.

Administratoren, die Messwerte aus ihren Servern auslesen und in die InfluxDB bringen möchten, können auf ein weiteres Werkzeug der InfluxDB-Entwickler zurückgreifen: „Telegraf“ ist eine handliche, in Go geschriebene Anwendung, die zum Beispiel die Prozessor- und Arbeitsspeicherauslastung ausliest und an die Datenbank meldet – die Installationsanleitung finden Sie über ct.de/yyyw. über Plug-ins ist die Open-Source-Software Telegraf leicht erweiterbar.

Rund um InfluxDB gibt es noch andere kompatible Open-Source-Projekte anderer Hersteller. Freunde ansprechender und funktionaler Diagramme sollten sich die Visualisierungsplattform Grafana ansehen, die Messwerte einer InfluxDB schnell und schön darstellen kann.

(jam@ct.de) c't

***Dokumentation:* ct.de/yyyw**