RDBエンジニアから見たInfluxDB
― 時系列DBってRDBとはどう違うの？

SRA OSS, Inc. 日本支社　近藤雄太

自己紹介

近藤雄太

SRA OSS, Inc. 日本支社に勤務

主な業務はPostgreSQL  の技術コンサル、サポート
最近は昨今の需要に伴い別種DBの検討、調査も

→InfluxDB  の調査で得られた知見が本日のお話

本セッションの目的

- 時系列DBとRDBとの違いを理解
- InfluxDBの概要を理解
- InfluxDBの有利なユースケースを理解

本セッションの主な対象者

RDBの心得があり、InfluxDBや
時系列DBをこれから知りたい方

目次

1.時系列DBとは
2.InfluxDBとは
3.InfluxDBを触ってみる
4.InfluxDBの特徴
5.InfluxDBの今後

時系列DBとは ー 定義

時系列DBとは・・・時系列データを扱うデータベース

時系列データとは・・・
ある現象において、時間が経過するにつれて変化する一連の値

時系列データの例としては、
気温、株価、交通量、各種センサー値

データベース的には、
時刻情報を必ず伴うデータで、
時刻から紐付いた値を取得できる
ものとして実装

時系列DBとは ー 利用目的

時系列DBに求められる要素

- 時系列に沿ったデータを効率な操作・管理
- 秒単位以下で読み書き可能なパフォーマンス
- リアルタイムデータ解析

時系列DBが実際に利用されている例

- IoT センサーデータ管理
- 金融データ解析

InfluxDBとは

1.時系列DBとは

2.InfluxDBとは

3.InfluxDBを触ってみる

4.InfluxDBの特徴

5.InfluxDBの今後

InfluxDB 基本情報

ライセンス	MIT (ただし、クラスタ機能は商用)
開発元	InfluxData主導,コントリビュータは400名程度
実装言語	Go
初版のリリース日	2016-09-08
バージョン	1系が主流、2系が最近リリース

時系列DBで人気1位 (db-engines.com調べ)

Rank Nov 2020	Rank Oct 2020	Rank Nov 2019	DBMS	Database Model	Score Nov 2020	Score Oct 2020	Score Nov 2019
1.	1.	1.	InfluxDB	Time Series	24.96	+0.81	+5.02
2.	2.	2.	Kdb+	Time Series, Multi-model	7.54	-0.12	+2.25
3.	3.	3.	Prometheus	Time Series	5.69	+0.36	+2.05

InfluxDBを触ってみる

1.時系列DBとは
2.InfluxDBとは
3.InfluxDBを触ってみる
4.InfluxDBの特徴
5.InfluxDBの今後

簡単InfluxDB構築

Docker HubにInfluxDB公式イメージあり

Dockerコマンドで即利用可能

```
# docker run --name=influxdb -d -p 8086:8086 influxdb
```

コマンドの意味は

- influxdbイメージを起動
- コンテナ名はinfluxdb (--name)
- コンテナ内部のポート8086をホストのポート8086に転送 (-p)
- デーモンとして起動 (-d)

InfluxDBに接続

- influxコマンドインタフェースを起動し、
  クエリを入力できる状態にする

```
# docker exec -it influxdb influx
>
```

- データベースを作成し、以降使用することを宣言

```
> CREATE DATABASE mydb
> USE mydb
```

次はデータを入力、とその前に

InfluxDBの扱うデータはRDBと以下の対応付けが可能

RDB

Table

	Column	Column	Column	Column	Column
Row					
Row					
Row					

InfluxDB

Measurement

	Time	TagA	TagB	FieldA	FieldB
Point					
Point					
Point					

InfluxDBを触ってみる ー データ構造(データ単位)

InfluxDBの扱うデータはRDBと以下の対応付けが可能

RDB

Table

	Column	Column	Column	Column	Column
Row					
Row					
Row					

RDBのデータは
行(row)が単位

InfluxDB

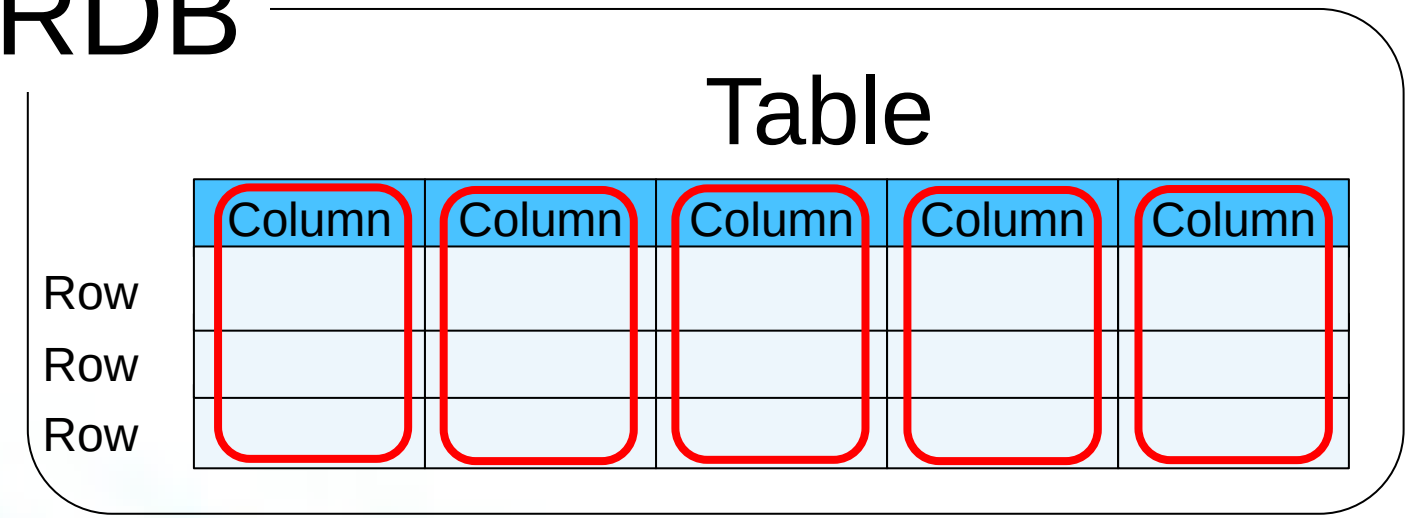
Measurement

	Time	TagA	TagB	FieldA	FieldB
Point					
Point					
Point					

InfluxDBのデータは
Pointが単位

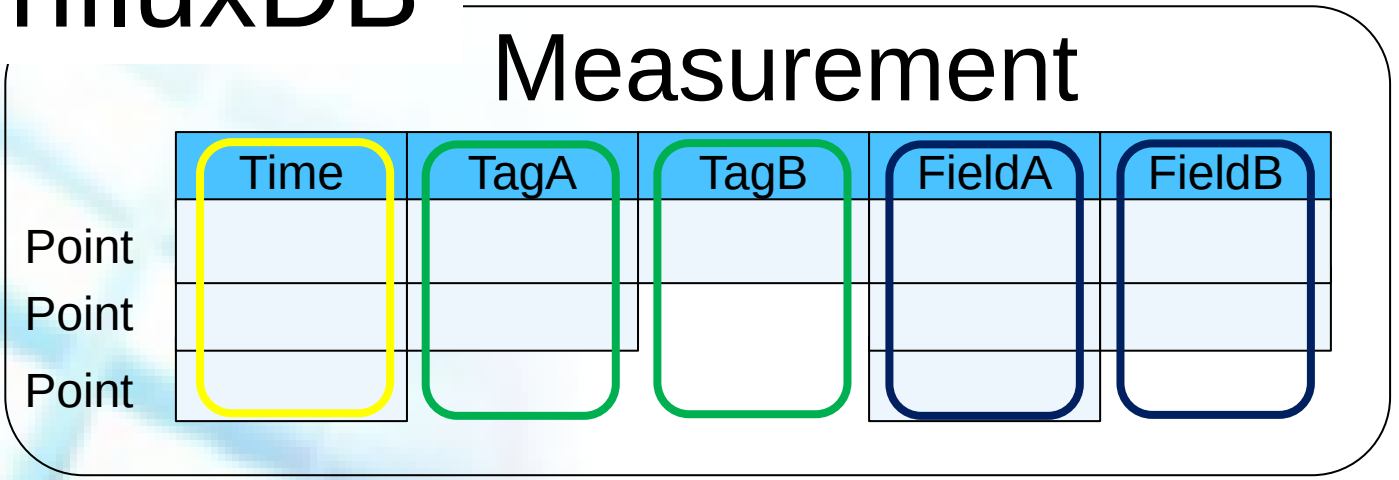
InfluxDBの扱うデータはRDBと以下の対応付けが可能

RDB



RDBのデータの
属性は列(column)

InfluxDB



InfluxDBのデータの
属性は3種類

Time:タイムスタンプ情報
Tag:インデックス用データ
Field:実際のデータ

InfluxDBの扱うデータはRDBと以下の対応付けが可能

RDB

Table

	Column	Column	Column	Column	Column
Row					
Row					
Row					

RDBのデータの
グループはテーブル(table)

InfluxDB

Measurement

	Time	TagA	TagB	FieldA	FieldB
Point					
Point					
Point					

InfluxDBのデータの
グループはMeasurement

※ただし、実際は
MeasurementはPointに含まれる要素です

SQLと似た書式のクエリ

データの入力にはINSERT, 参照にはSELECT

- データを入力

```
> INSERT cpu,host=serverA,region=Japan value=0.64
```

Measurement, Tag, Fieldの順に記載

- データを参照

```
> SELECT * FROM cpu
name: cpu
time                           host    region value
----                           ----    ------ -----
2020-06-20T13:12:54.391399679Z serverA Japan  0.64
```

InfluxDBの特徴

1.時系列DBとは
2.InfluxDBとは
3.InfluxDBを触ってみる
4.InfluxDBの特徴
5.InfluxDBの今後

InfluxDBの特徴 — CRUD

RDBはCRUD、InfluxDBはCR-ud

時系列データは一度書き込まれたデータを更新することが稀

データの更新(Update)、削除(Delete)を意図的に制限し、
作成(Create)、読み取り(Read)のパフォーマンスを優先する
設計

データの更新について

UPDATE文は用意されていない
同一のタイムスタンプとタグを持ったデータをINSERTすることで更新される

タイムスタンプ、タグの更新は不可能

```
> SELECT * FROM cpu2
name: cpu2
time                host    region value
----                ----    ------ -----
1614763935044132135 serverA Japan  0.64
> INSERT cpu2,host=serverA,region=Japan value=0.75
1614763935044132135
> SELECT * FROM cpu2
name: cpu2
time                host    region value
----                ----    ------ -----
1614763935044132135 serverA Japan  0.75
```

データの削除について

SQLと同様にDELETE文で削除可能
ただし、WHERE句にフィールド値を指定不可能

他に、リテンションポリシーにより一括削除がある（後述）

```
> SELECT * FROM cpu2
name: cpu2
time                host    region value
----                ----    ------ -----
1614763935044132135 serverA Japan  0.75
> DELETE FROM cpu2 where value = 0.75;
ERR: shard 3: fields not supported in WHERE clause during deletion
> DELETE FROM cpu2 where host = 'serverA';
> SELECT * FROM cpu2;
>
```

InfluxDBの特徴 ― クエリ言語

RDBはSQL、InfluxDBは独自のクエリ言語

SQLの構文に似せた「InfluxQL」

```
SELECT * FROM “cpu" WHERE time > now() - 1h
```

関数型言語パターンの「Flux」

```
from(bucket: “…”)
  |> range(start:-1h)
  |> filter(fn: (r) => r._measurement == “cpu”
```

Fluxの機能的メリット

- Tagを用いた並び替えのサポート
- 結合（JOIN）操作のサポート

など

InfluxDBの特徴 ー トランザクション（ACID特性）

RDBにはトランザクションがあり、
InfluxDBには**トランザクションがない**

トランザクションによるオーバヘッドを避けることで**パフォーマンス向上↑**
しかし、ACID特性を保証されることが前提のユースケースでは使えない

ACID特性	RDB	InfluxDB	特性の説明
原子性 Atomicity	○	×	処理は完全に実行されるか全く実行されないのどちらかとなる
一貫性 Consistency	○	×	処理結果に依らず、データベースの整合性が保たれる
独立性 Isolation	○	×	複数の処理を実行しても単独に実行した場合と同じ処理結果
耐久性 Durability	○	○ (WALにより実現)	障害が発生してもデータが失われない

InfluxDBの特徴 ― 時系列データ管理

RDBのデータ管理は汎用的、
InfluxDBのデータ管理は**時系列データ特有**

短時間に大量のデータが保存される特性に合わせて
以下の機能が標準搭載

- データ圧縮
- 古くなった生データの定期削除
- 生データの要約

InfluxDBの特徴 ― 時系列データ管理（データ定期削除）

リテンションポリシー（Retention Policy）

データの保持期間設定をデータベース内に作成

メジャーメントはいずれかのリテンションポリシーに属する
デフォルトでは、保持期間が無限の”autogen”という名前のポリシーに所属する

データを12時間分（1時間毎に1時間分削除）保持するポリシー

```
> CREATE RETENTION POLICY policy1 ON mydb DURATION 12h SHARD DUPLICATION 1h REPLICATION 1
> INSERT INTO policy1 measurement1 tag1=123, field1=456
```

データベース作成時にデフォルトのポリシーを設定することも可能

```
> CREATE DATABASE mydb WITH DURATION 12h SHARD DUPLICATION 1h REPLICATION 1 NAME policy_1
```

InfluxDBの特徴 ― 時系列データ管理(データの要約)

コンティニュアスクエリ(Continuous Query)

定期的にクエリを実行(クエリ結果をメジャーメントに格納)するもので、
生データを要約してデータ量を減らすことを目的に利用

GROUP BY time()で設定した時間ごとに実施される

30分間の平均値を残すためのコンティニュアスクエリ

```
CREATE CONTINUOUS QUERY continuous_query_1
ON mydb
BEGIN
  SELECT mean(value) INTO average_cpu FROM cpu
    GROUP BY time(30m), region
END
```

InfluxDBの特徴 ー データ可視化

Chronograf

InfluxDB 1.xプラットフォームで
用意されたユーザインタフェース

Grafana

InfluxDBを含む16種のデータ
ソースを公式サポート(v7.3現在)

簡単Chronograf構築

Docker HubにChronograf公式イメージあり

influxDBイメージと連携してすぐにChronografを利用可能

コンテナ間ネットワーク作成

```
# docker network create influxdb
```

influxdbコンテナ起動（先に起動したinfluxdbコンテナは削除しておく）

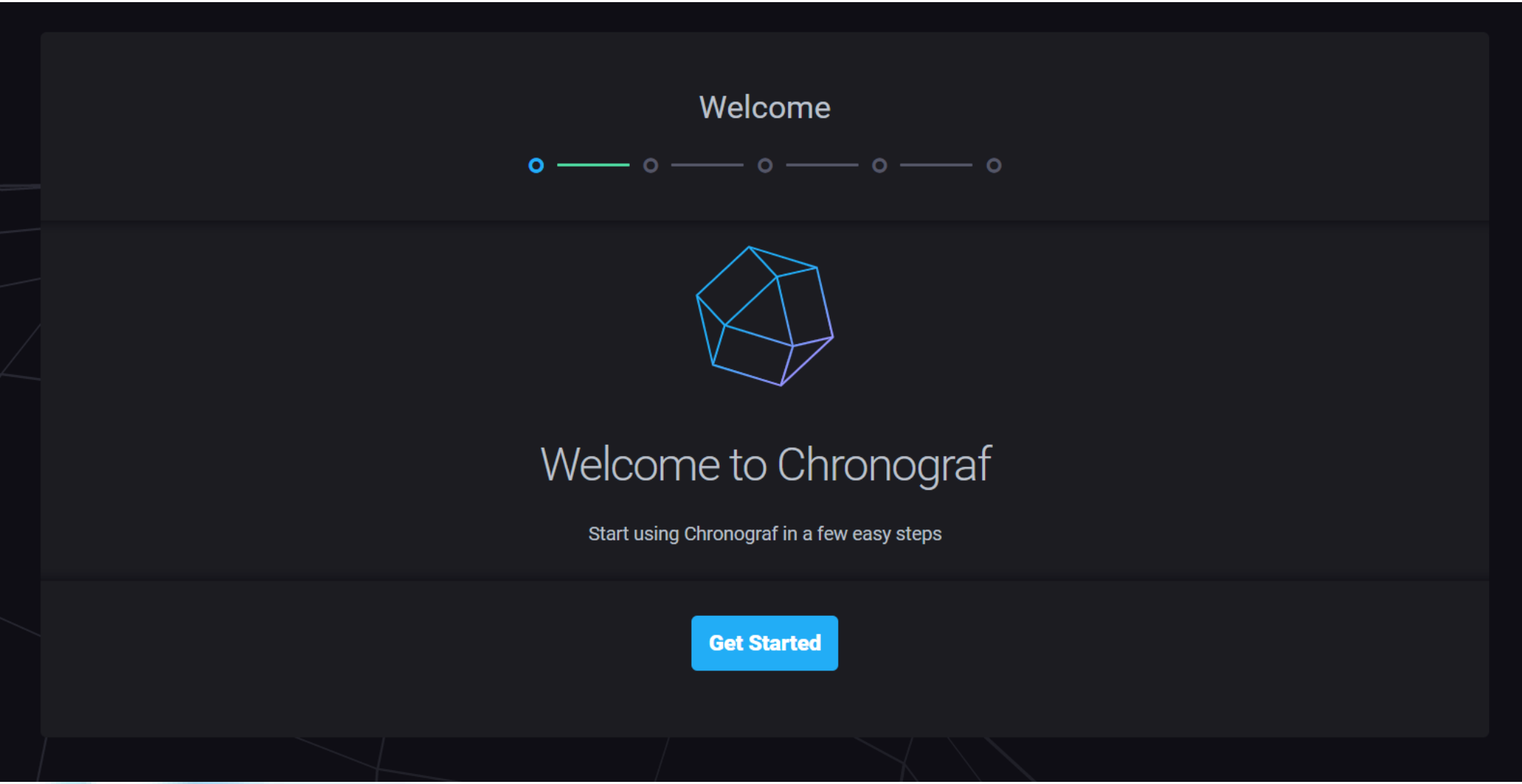
```
# docker run --name=influxdb -d -p 8086:8086 --net=influxdb influxdb
```

chronografコンテナ起動

```
# docker run -p 8888:8888 --influxdb-url=http://influxdb:8086 chronograf
```

InfluxDBの特徴 ― データ可視化(Chronograf)

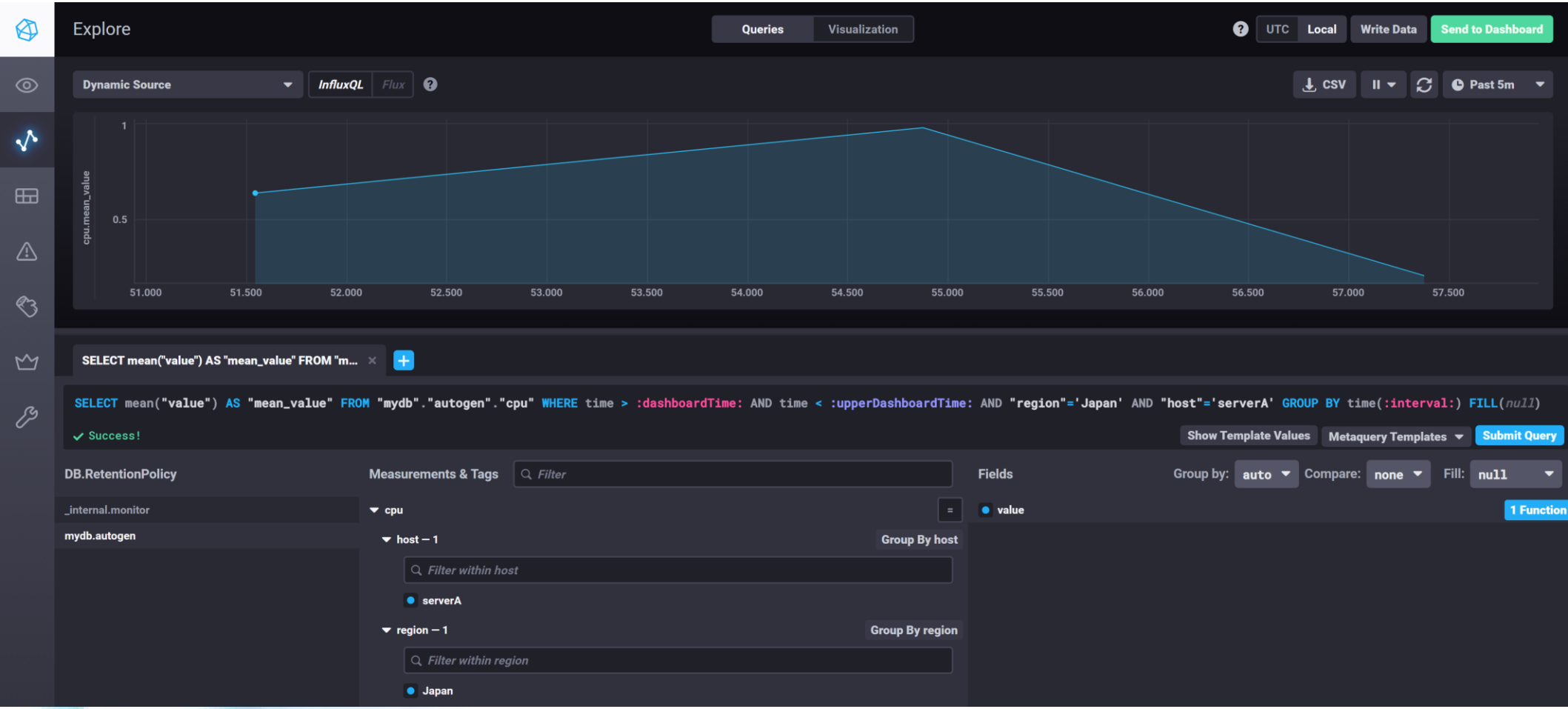
docker コンテナを起動したホスト（ポート:8888）に
Webブラウザでアクセスすると、



InfluxDBの特徴 ― データ可視化(Chronograf)

データを入力すると、

```
> INSERT cpu,host=serverA,region=Japan value=0.64
> INSERT cpu,host=serverA,region=Japan value=0.98
> INSERT cpu,host=serverA,region=Japan value=0.21
```



InfluxDBの特徴 ― 時系列データ解析

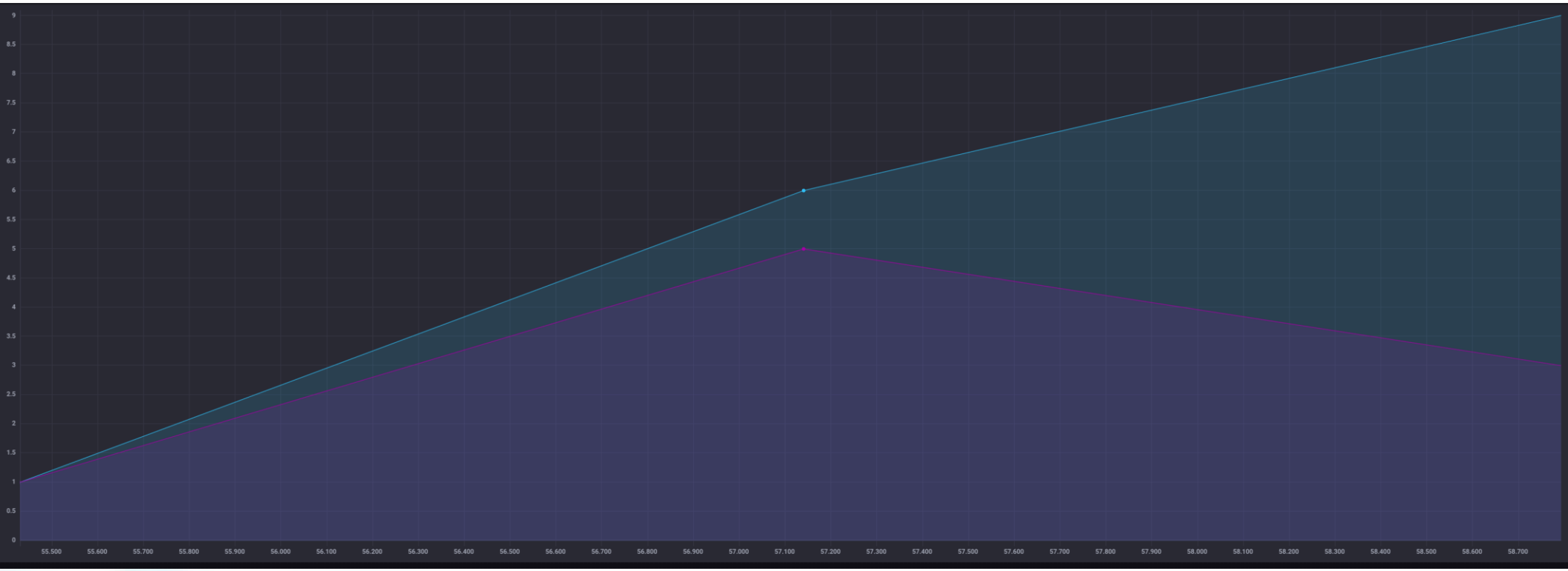
RDBは汎用的な数学関数のみサポート

InfluxDBはさらに**時系列データ解析用関数**をサポート

InfluxDBのサポートする解析用関数の一部

- 累積合計
- ヒストグラム
- 各種移動平均
- パーセンタイル
- サンプル抽出
- 時空間データ用計算

累積合計 cumulative_sum()

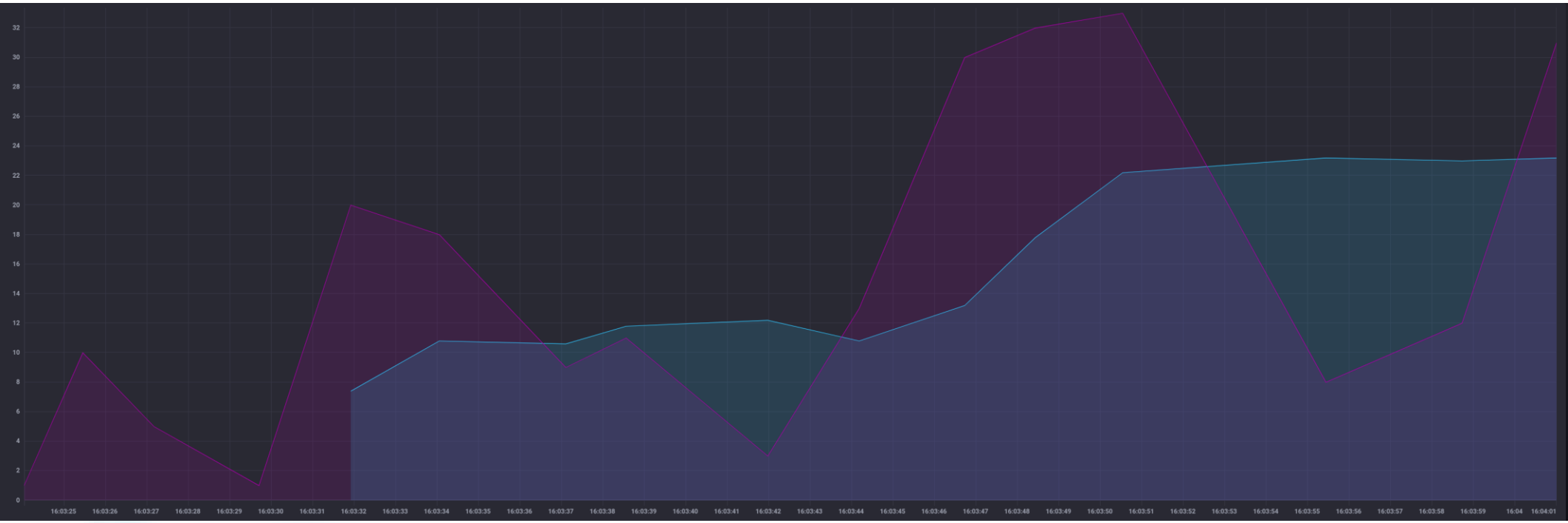


SELECT value FROM mydb.autogen.cum
→ 1, 5, 3

SELECT cumulative_sum(value) FROM mydb.autogen.cum
→ 1, 6(=1+5), 9(=1+5+3)

InfluxDBの特徴 ― 時系列データ解析（移動平均）

移動平均 moving_average()



SELECT value FROM mydb.autogen.mov
→ 1, 10, 5, 1, 20, 18, ..

SELECT moving_average(value, 5) FROM mydb.autogen.mov
→ 7.4 (= (1+10+5+1+20)/5), 10.8 (= (10+5+1+20+18)/5) , ..

InfluxDBの特徴 ー まとめ

InfluxDBは汎用的なRDBに比べて、

時系列データを用いるユースケースに
特化することで強力なパフォーマンスを発揮

特に本セッションの対象者の方へ

一般的なRDBの常識が通じない点が多く、InfluxDBを使用する際は「利用目的の理解」と「InfluxDBの十分な学習」が必須

→別RDBを覚えるよりも学習コスト大

InfluxDBの今後

1.時系列DBとは
2.InfluxDBとは
3.InfluxDBを触ってみる
4.InfluxDBの特徴
5.InfluxDBの今後

InfluxDBの今後 ― InfluxDB 2.0

InfluxDB 2.0 GAが2020-11-10にリリースされました

- クエリ言語がInfluxQLからFluxがメインに
- データ解析・可視化構成をテンプレートとして作成可能に
  - 設定項目はデータソース、解析操作、ダッシュボード、アラートなど
  - 様々なユースケースにおける構成を共通化することでセットアップ時間の短縮に
- 1.xからのデータ移行は容易

InfluxDBの今後 ー InfluxDB IOx（アイオックス）

新コアエンジンを開発中

- Tag値の種類が多いデータにおける
  パフォーマンス劣化を解消
- SQLをサポート

- 2021年後半リリース予定
- 後方互換性を維持

ご清聴ありがとうございました