Easy Solutions

Big Data mit InfluxDB

beer + code #13

beer + code ried

Vorstellung

Wendling

Robert Wurm

Unsere Anforderungen an Big Data

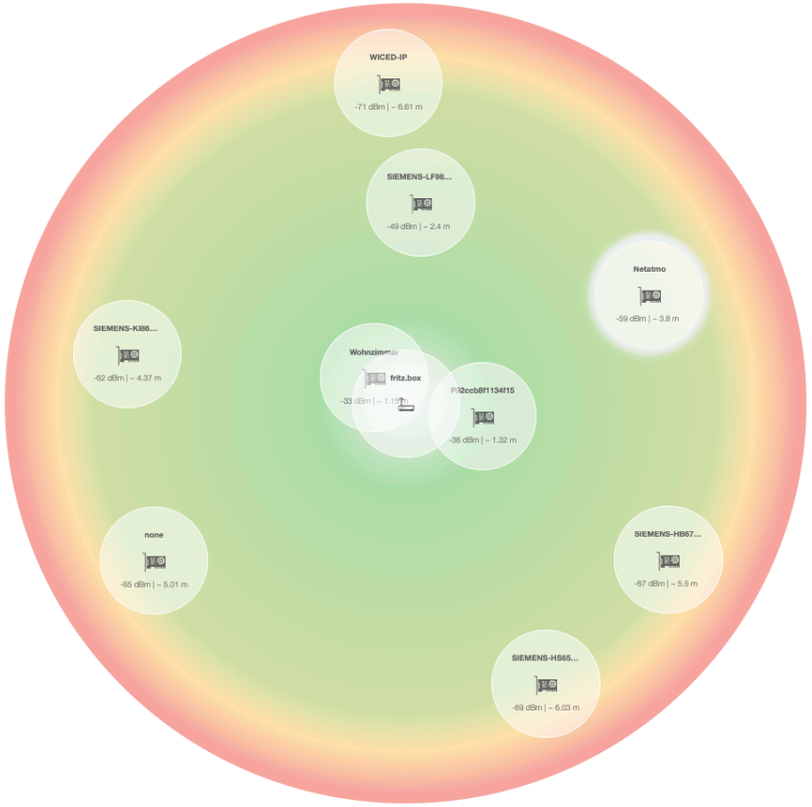
Endgeräte

Daten

Router

WLAN **LAN** **IP**

Telefonie **WAN** **Traffic**

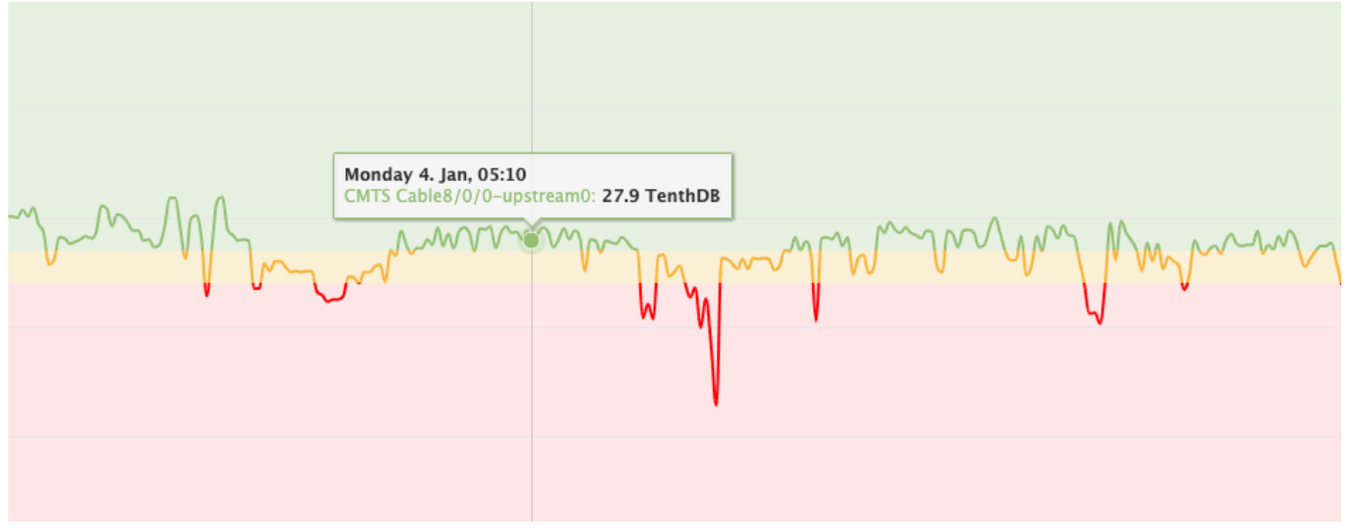


Daten

Router

WLAN LAN IP

Telefonie WAN Traffic



Kabelmodem

HF-Werte **Signalqualität**

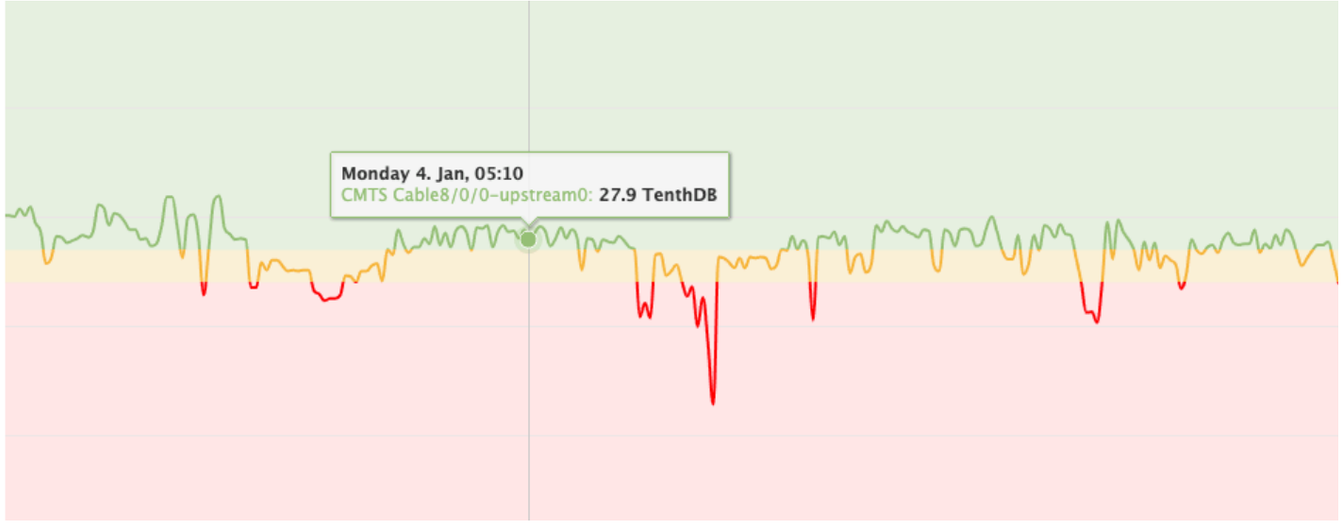
Upstream **Downstream**

Daten

Router

WLAN LAN IP

Telefonie WAN Traffic



Kabelmodem

HF-Werte Signalqualität

Upstream Downstream

Infrastruktur

Bandbreite LWL-Werte

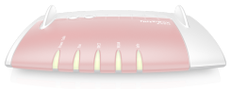
Upstream Downstream

Daten

Router

WLAN LAN IP

Telefonie WAN Traffic



Kabelmodem

HF-Werte Signalqualität

Upstream Downstream

Infrastruktur

Bandbreite LWL-Werte

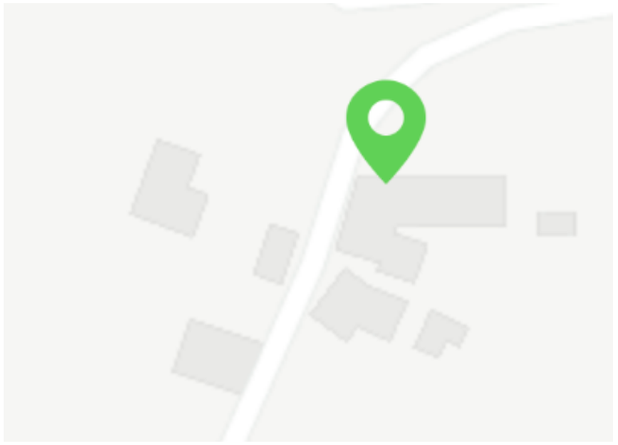
Upstream Downstream

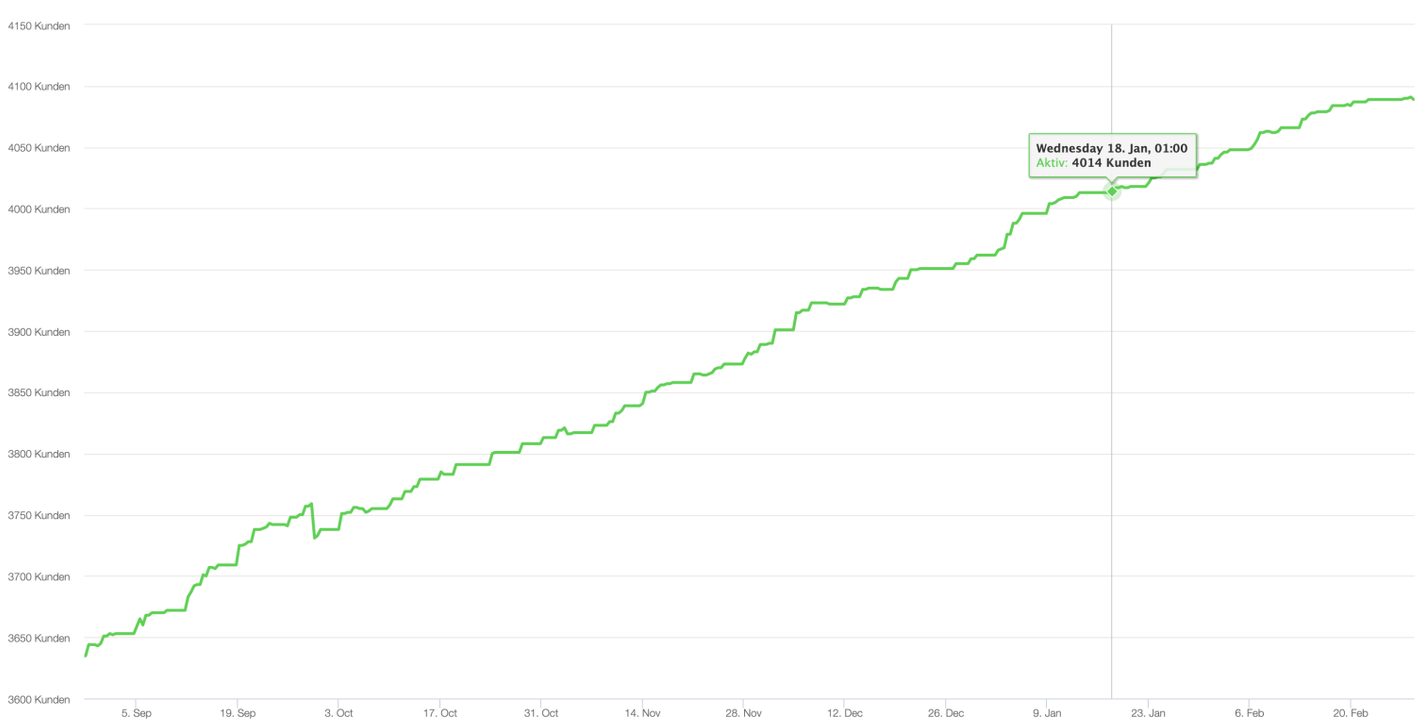
Endkunden

Name **Adresse**

Verträge **Bandbreite**

Nr.:
205584
Erstellt:
15 Tage
Adresse:
Pauredt 8A - 4741 Wendling
E-Mail:
robert.wurm@easysol.com
Telefon Nr.:
+436641059402



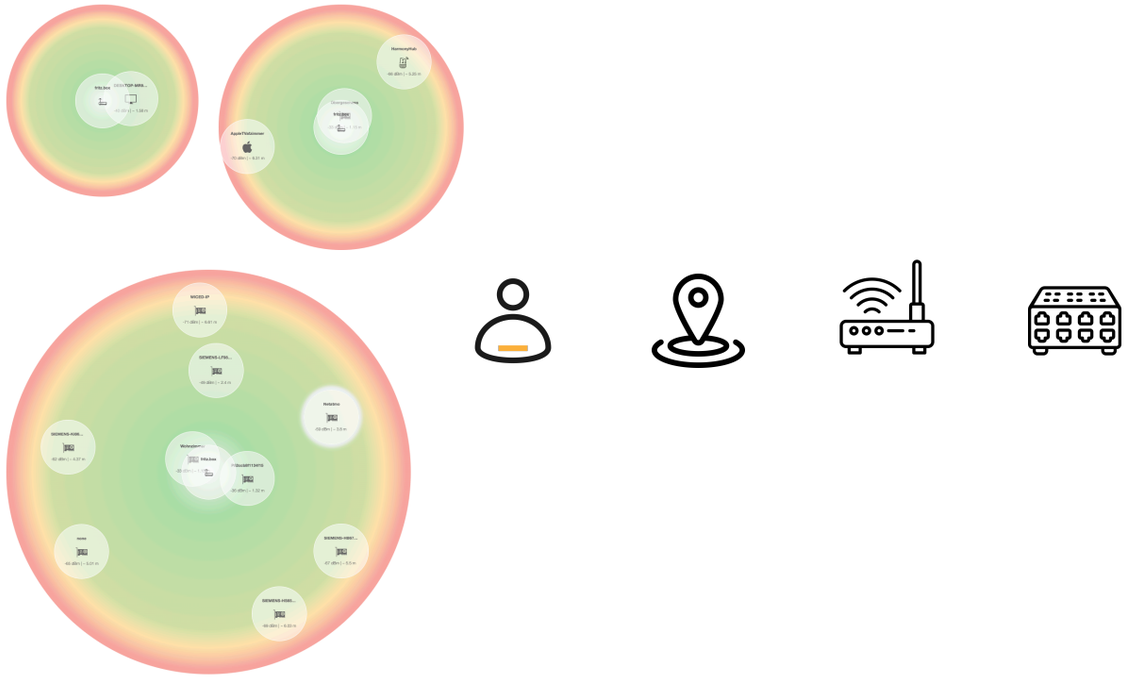


Anforderungen

Verschiedene Typen von Daten

Relationale Daten **Nicht Relationale Daten**

Zeitbasierte Daten



Anforderungen

Verschiedene Typen von Daten

Relationale Daten Nicht Relationale Daten

Zeitbasierte Daten

Flexible Datenerfassung

TR-069 **TR-369** **SNMP**

SQL-Statements

Anforderungen

Verschiedene Typen von Daten

Relationale Daten Nicht Relationale Daten

Zeitbasierte Daten

Flexible Datenerfassung

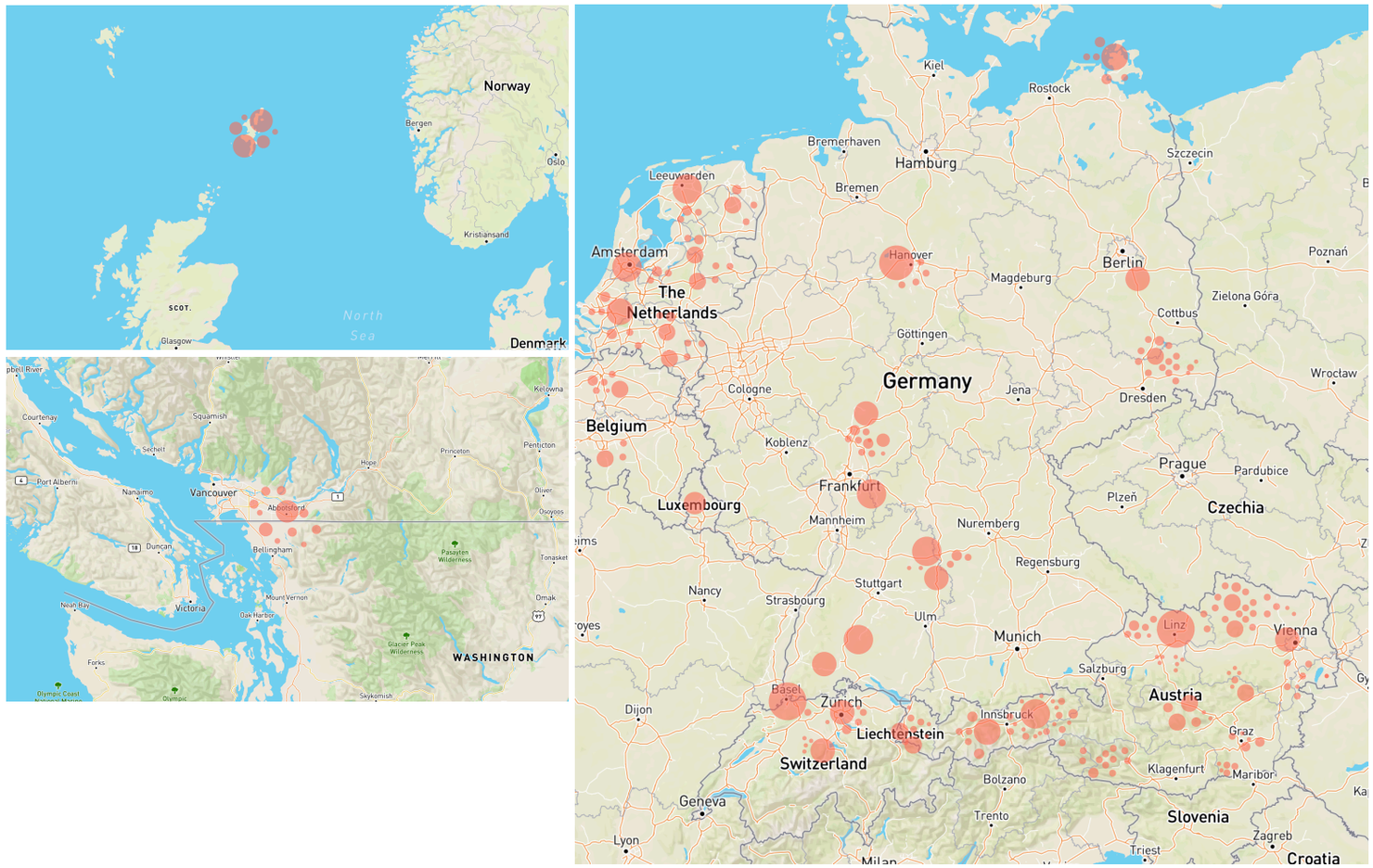
TR-069 TR-369 SNMP

SQL-Statements

Effiziente Datenpersistierung

Speicher Abfragezeiten

Dynamik



Anforderungen

Verschiedene Typen von Daten

Relationale Daten Nicht Relationale Daten

Zeitbasierte Daten

Flexible Datenerfassung

TR-069 TR-369 SNMP

SQL-Statements

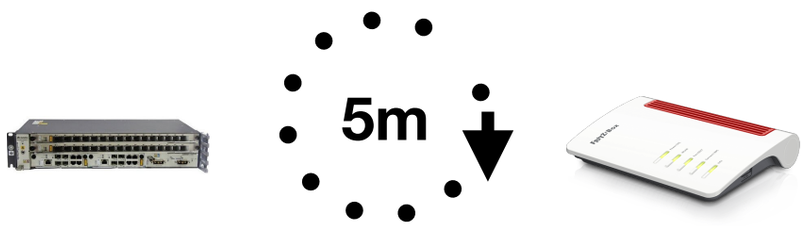
Effiziente Datenpersistierung

Speicher **Abfragezeiten**

Dynamik

~ 250.000 Geräte

pro Gerät ~ 500 Datenpunkte



~125.000.000 Datenpunkte alle 5 Minuten

~ 8 Bytes pro Datenpunkt

~ 1 GB alle 5 Minuten

~ 288 GB jeden Tag

Anforderungen

Verschiedene Typen von Daten

Relationale Daten **Nicht Relationale Daten**

Zeitbasierte Daten

Flexible Datenerfassung

TR-069 **TR-369** **SNMP**

SQL-Statements

Effiziente Datenpersistierung

Speicher **Abfragezeiten**

Dynamik

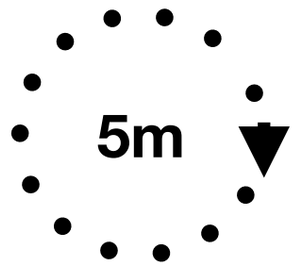
ca. **105 TB** im Jahr

DB
Performance Vergleiche

Testaufbau

Attribut	Datentype	Beschreibung
time	Timestamp	Zeitpunkt der Messung.
host	Text	Hostname des CPEs an dem die Messung durchgeführt wurde.
ifName	Text	Name des Kanals der Messung.
docsIfCmStatusTxPower	Number	Gemessene Upstream Kanal-Sendeleistung.
docsIfSigQMicroreflections	Number	Gemessene Mikroreflexionen am Downstream Kanal.
docsIfSigQSignalNoise	Number	Signal-Rausch-Verhältnis am Downstream Kanal.
docsIfDownChannelPower	Number	Gemessene Kanalleistung am Kabel-Modem.
docsIfSigQCorrecteds	Counter32	Zählerstand der empfangenen Codewords (IP-Datenpackete) mit Fehler die jedoch korrigiert werden konnten.
docsIfSigQUnerroreds	Counter32	Zählerstand der empfangenen Codewords ohne Fehler.
docsIfSigQUncorrectables	Counter32	Zählerstand der unkorrigierbaren empfangenen Codewords.
docsIfDownChannelFrequency	Number	Die Frequenz des Downstream-Kanals.
docsIfDownChannelModulation	Number	Die angewendete Kanal Modulation für die Übertragung der Daten.
docsIf3RxChStatusPrimaryDsIndicator	Boolean	Deklariert den Downstream-Kanal als Primary-Channel.

190 Kabelmodems
13 Attribute

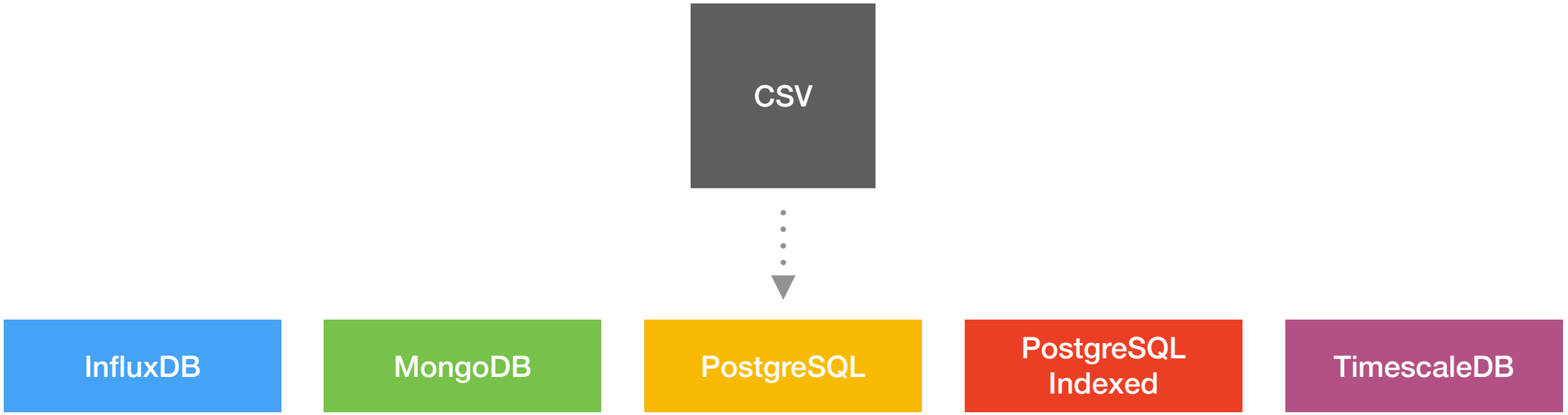


SNMP

6,06 GB

Zeitraum von 2 Monaten (01. Mai 2019 bis 01. Juli 2019) von ca. 190 Kabel-Modems in eine CSV-Datei exportiert.

Testaufbau


CSV
InfluxDB
MongoDB
PostgreSQL
PostgreSQL Indexed
TimescaleDB

Messungen

MacBook Pro mit 16GB LPDDR3 RAM und 3,5 GHz Intel Core

Laufzeit Import

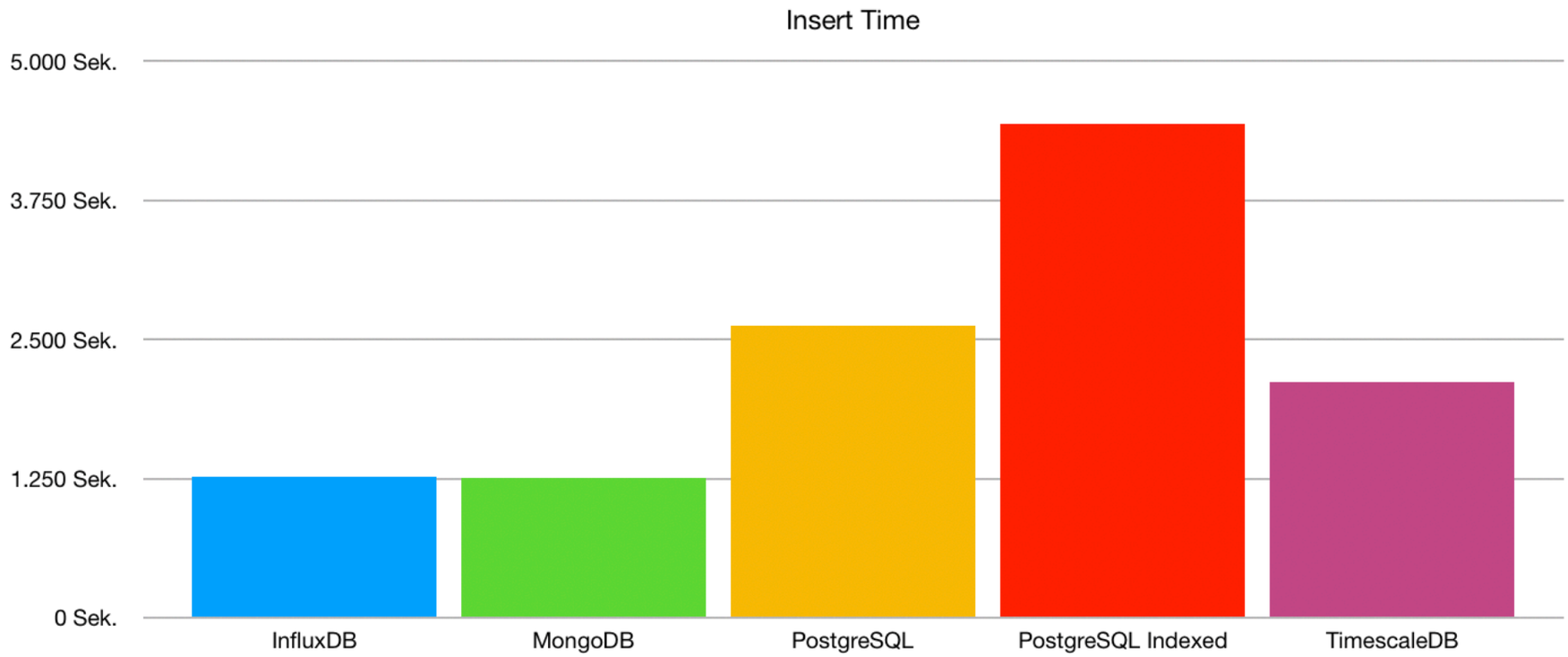
Abfrage-Operationen

Datenbankgröße

Messungen

MacBook Pro mit 16GB LPDDR3 RAM und 3,5 GHz Intel Core

Laufzeit Import



Insert Runtime

	InfluxDB	MongoDB	PostgreSQL	PostgreSQL Indexed	TimescaleDB
Time (sek.)	21 Min.	21 Min.	44 Min.	74 Min.	35 Min.

Messungen

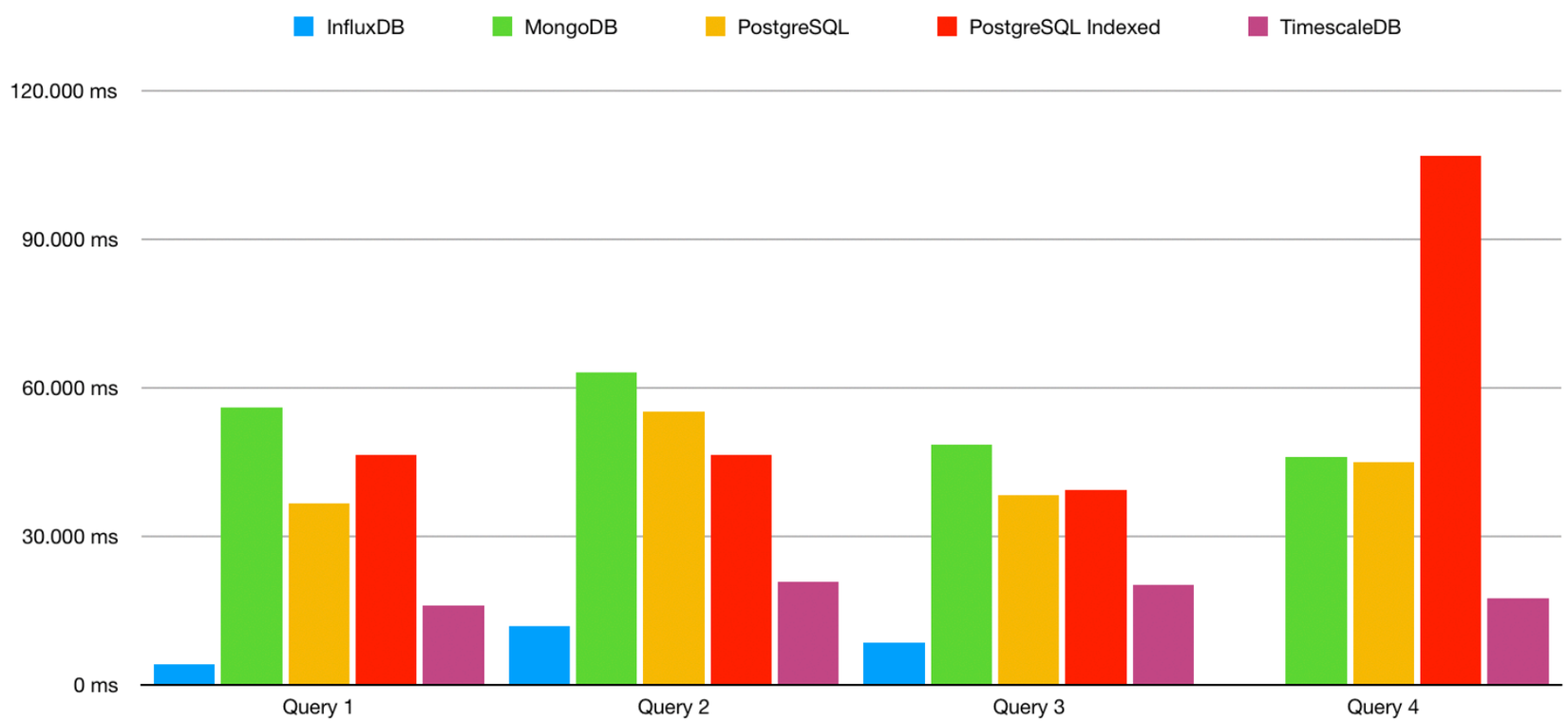
MacBook Pro mit 16GB LPDDR3 RAM und 3,5 GHz Intel Core

Abfrage-Operationen

```
/* Query 1 */
SELECT mean("docsIfSigQSignalNoise")
    FROM "cm_downstream"

/* Query 2 */
SELECT mean("docsIfSigQSignalNoise")
    FROM "cm_downstream"
    GROUP BY "host"

/* Query 3 */
SELECT max("docsIfSigQSignalNoise")
    FROM "cm_downstream"
    WHERE "docsIfSigQSignalNoise" < 250
    GROUP BY "host"
```



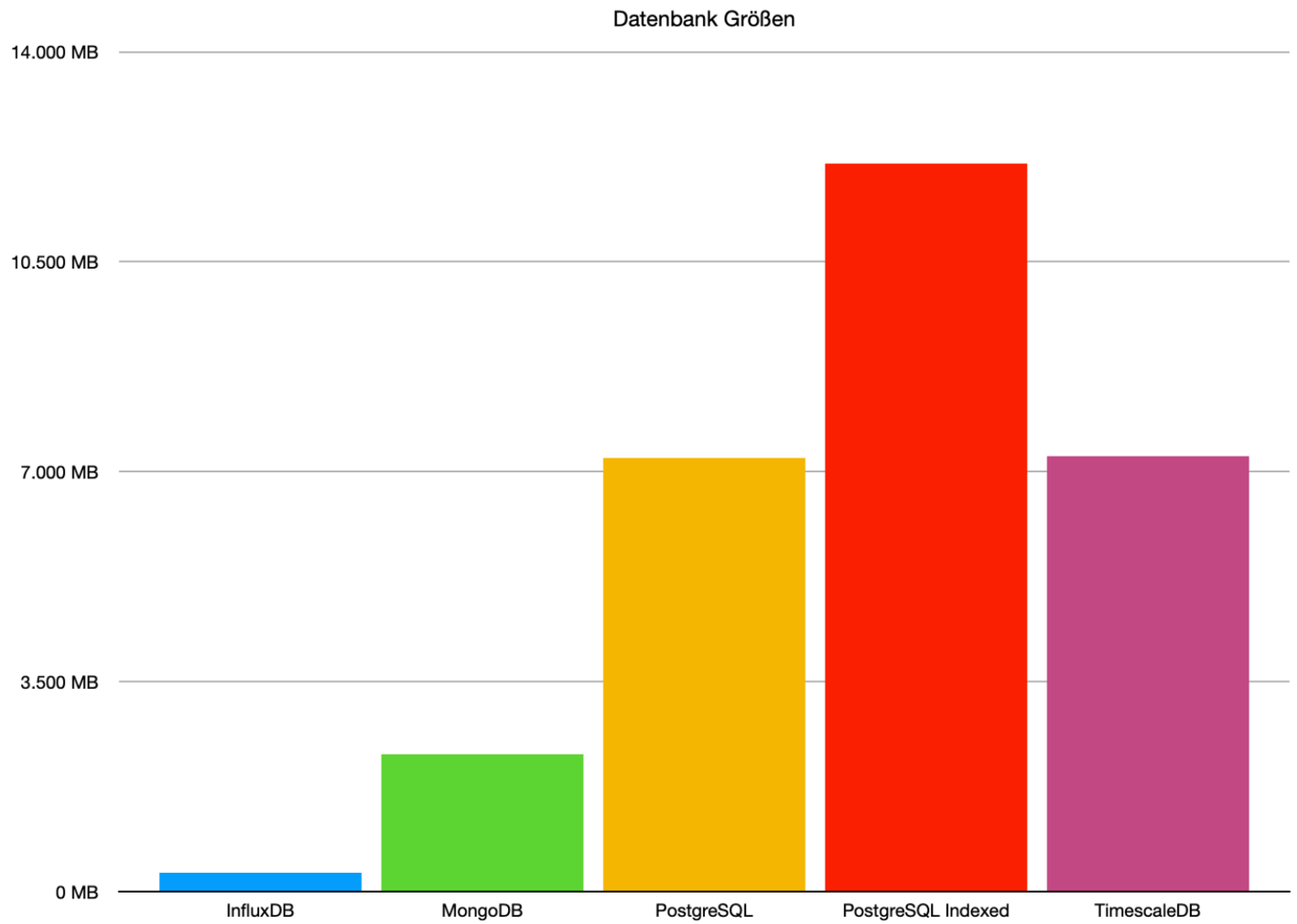
Query Runtime

	InfluxDB	MongoDB	PostgreSQL	PostgreSQL Indexed	TimescaleDB
Query 1	4.209 ms	56.075 ms	36.639 ms	46.466 ms	16.006 ms
Query 2	11.935 ms	63.080 ms	55.300 ms	46.399 ms	20.934 ms
Query 3	8.617 ms	48.548 ms	38.290 ms	39.392 ms	20.212 ms
Query 4	101 ms	46.051 ms	45.017 ms	106.947 ms	17.532 ms

Messungen

MacBook Pro mit 16GB LPDDR3 RAM, einem 3,5 GHz Intel Core

Datenbankgröße



DB Size

	InfluxDB	MongoDB	PostgreSQL	PostgreSQL Indexed	TimescaleDB
Size (MB)	314 MB	2.297 MB	7.235 MB	12.131 MB	7.262 MB

InfluxDB

InfluxDB

- Open Source TSDB (MIT); Released 2013
- Implementiert in GO
- Sehr einfacher Einstieg - keine Dependencies
- SQL-like REST-API
- Gute CLI

Keine TSDB


The Da Vinci Code
The Green Mile
Cloud Atlas
Apollo 13
That Thing You Do
Cast Away
Tom Hanks
Joe Versus the Volc…
A League of Their Own
Sleepless
You've Got Mail
Charlie Wilson's War
The Polar Express
ACTED_IN
DIRECTED

TSDB

InfluxDB Model

database

```
$ influx -precision rfc3339
Connected to http://localhost:8086 version 1.8.10
InfluxDB shell 1.8.10
>
```

```
> CREATE DATABASE mydb
>
```

```
> SHOW DATABASES
name: databases
name
----
_internal
mydb

>
```

```
> USE mydb
Using database mydb
>
```

InfluxDB Model

database

measurement

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

```
> SELECT "host", "region", "value" FROM "cpu"
name: cpu
---------
time                           host      region   value
2015-10-21T19:28:07.580664347Z serverA   us_west  0.64

>
```

series

```
cpu,host=serverA,region=us_west value=0.64
payment,device=mobile,product=Notepad,method=credit billed=33,licenses=3i 14340674
stock,symbol=AAPL bid=127.46,ask=127.48
temperature,machine=unit42,type=assembly external=25,internal=37 14340674670000000
```

tag

indexed

`host` `region`

field

`value`

InfluxDB Model

database

measurement

series

tag

field

keywords

```
ALL          ALTER        ANY          AS           ASC          BEGIN
BY           CREATE       CONTINUOUS   DATABASE     DATABASES    DEFAULT
DELETE       DESC         DESTINATIONS DIAGNOSTICS  DISTINCT     DROP
DURATION     END          EVERY        EXPLAIN      FIELD        FOR
FROM         GRANT        GRANTS       GROUP        GROUPS       IN
INF          INSERT       INTO         KEY          KEYS         KILL
LIMIT        SHOW         MEASUREMENT  MEASUREMENTS NAME         OFFSET
ON           ORDER        PASSWORD     POLICY       POLICIES     PRIVILEGES
QUERIES      QUERY        READ         REPLICATION  RESAMPLE     RETENTION
REVOKE       SELECT       SERIES       SET          SHARD        SHARDS
SLIMIT       SOFFSET      STATS        SUBSCRIPTION SUBSCRIPTIONS TAG
TO           USER         USERS        VALUES       WHERE        WITH
WRITE
```

```
> SELECT * FROM /.*/ LIMIT 1
--
> SELECT * FROM "cpu_load_short"
--
> SELECT * FROM "cpu_load_short" WHERE "value" > 0.9
```

InfluxDB Model

database

measurement

series

tag

field

keywords

```
> SELECT MEAN("index") FROM "h2o_quality" GROUP BY "location","randtag"

name: h2o_quality
tags: location=coyote_creek, randtag=1
time                   mean
----                   ----
1970-01-01T00:00:00Z   50.69033760186263

name: h2o_quality
tags: location=coyote_creek, randtag=2
time                   mean
----                   ----
1970-01-01T00:00:00Z   49.661867544220485

name: h2o_quality
tags: location=coyote_creek, randtag=3
time                   mean
----                   ----
1970-01-01T00:00:00Z   49.360939907550076

name: h2o_quality
tags: location=santa_monica, randtag=1
time                   mean
----                   ----
```

InfluxDB Model

database

measurement

series

tag

field

keywords

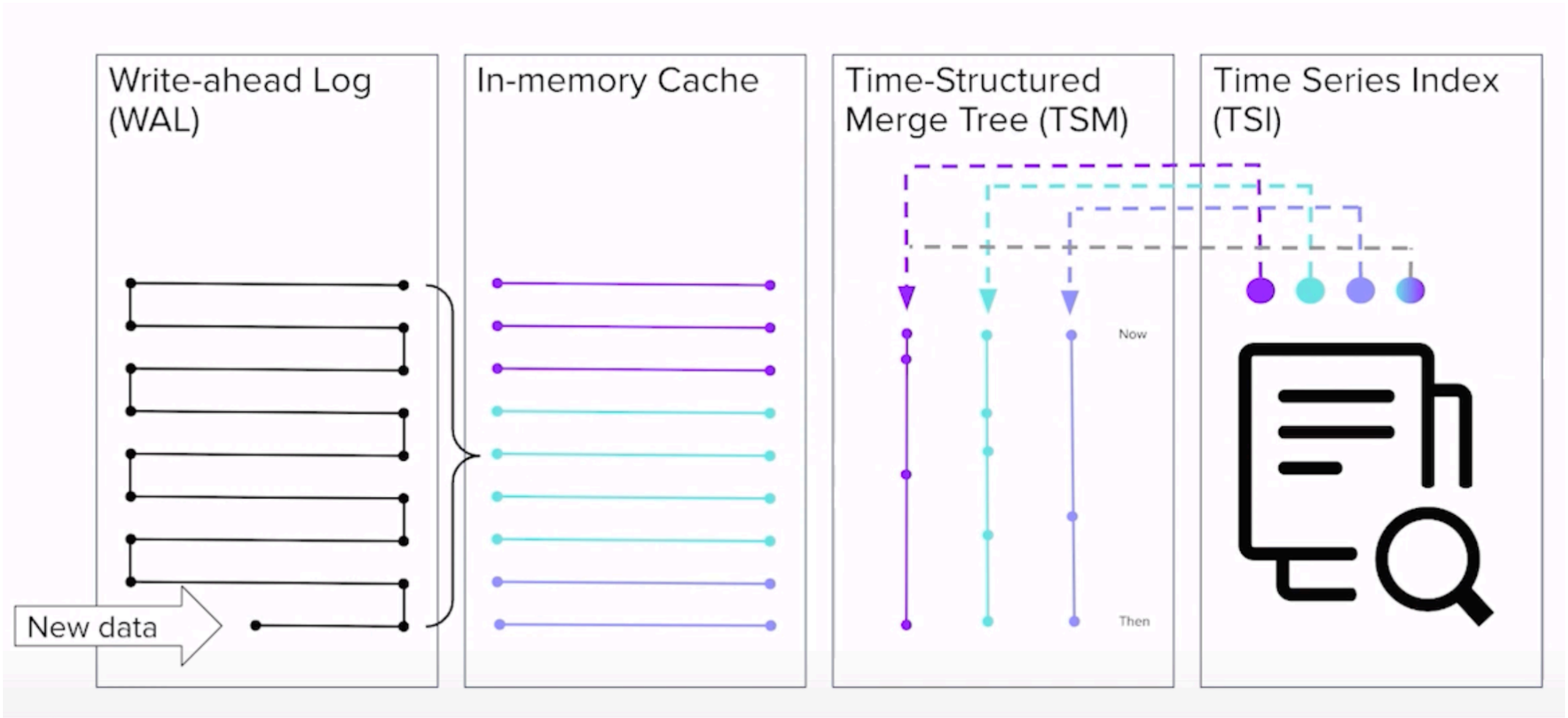
REST api

```
$ curl -G 'http://localhost:8086/query?db=mydb' --data-urlencode 'q=SELECT * FROM "mymeas"'

{"results":[{"statement_id":0,"series":[{"name":"mymeas","columns":["time","myfield","mytag1","mytag2"],"
```

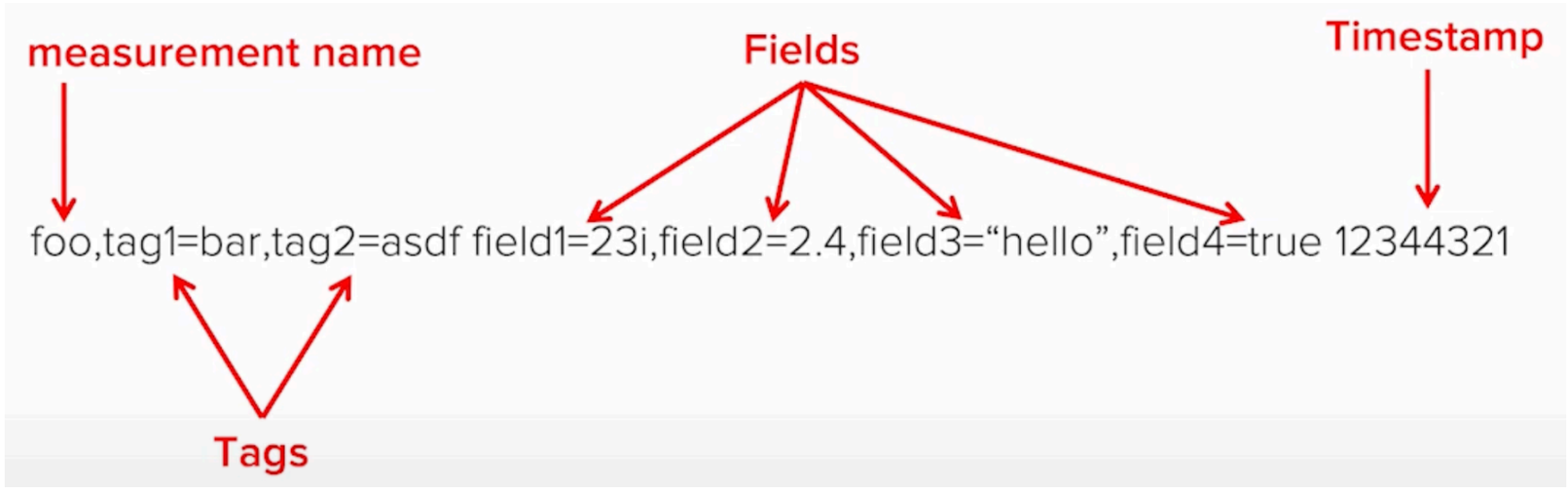
InfluxDB Model

select value **from** response_times **where** time > now() -1h **limit** 1000;



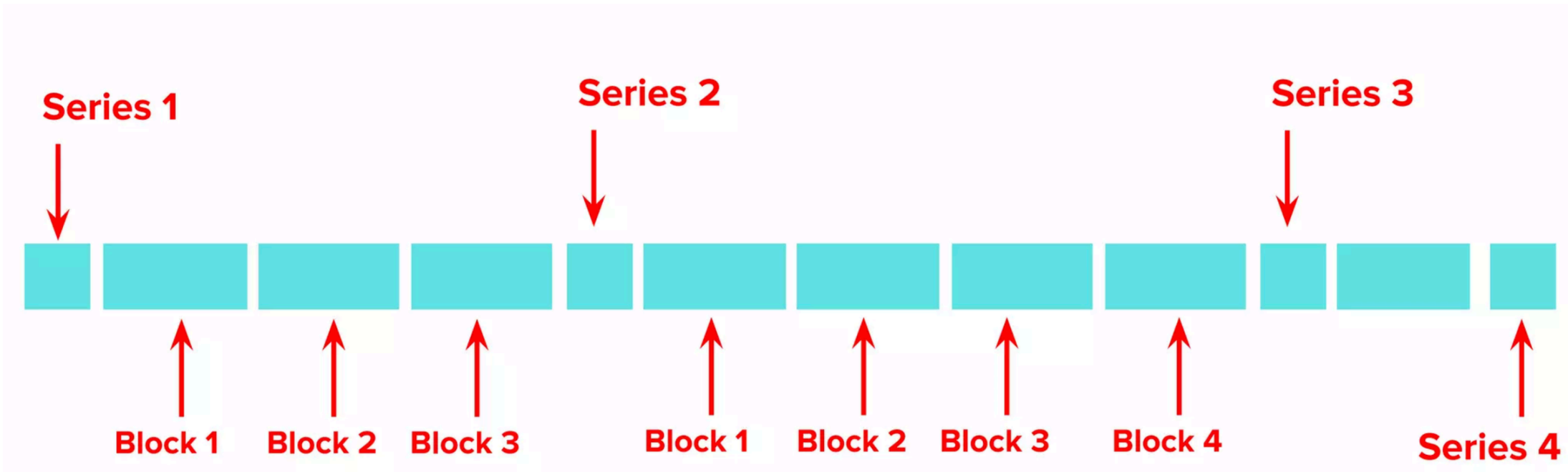
Time-Structured Merge Tree (TSM)

line protocol



Time-Structured Merge Tree (TSM)

TSM Data (files, storage engine)



Anstatt einzelner Werte -> Blocks

Time value pairs

Time-Structured Merge Tree (TSM)

Inverted Index

Line Protocol

```
weather,city=Denver,state=CO,zip=80222 temp=62.3,humidity=32
weather,city=Bellevue,state=WA,zip=98007 temp=50.7,humidity=76
weather,city=Brooklyn,state=NY,zip=11249 temp=58.2,humidity=55
```

Series

```
weather,city=Denver,state=CO,zip=80222#temp – 1
weather,city=Denver,state=CO,zip=80222#humidity - 2
weather,city=Bellevue,state=WA,zip=98007#temp - 3
weather,city=Bellevue,state=WA,zip=98007#humidity – 4
weather,city=Brooklyn,state=NY,zip=11249#temp – 5
weather,city=Brooklyn,state=NY,zip=11249#humidity - 6
```

Organisation der Meta-Daten

Time-Structured Merge Tree (TSM)

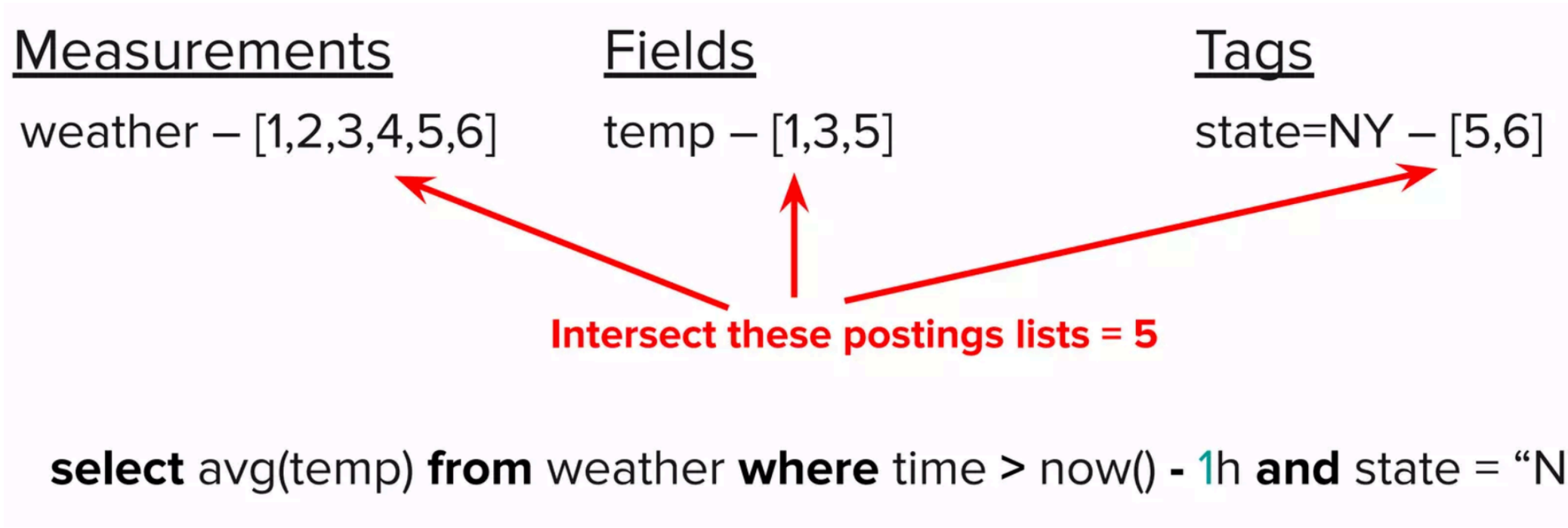
Posting List

Measurements	Fields	Tags
weather – [1,2,3,4,5,6]	temp – [1,3,5] humidity – [2,4,6]	city=Denver – [1,2] city=Bellevue – [3,4] city=Brooklyn – [5,6] state=CO – [1,2] state=NY – [5,6] state=WA – [3,4] zip=11249 – [5,6] zip=80222 – [1,2] zip=98007 – [3,4]

select temp **from** weather **where** time **>** now() **-** 1h **and** state = “NY”;

Time-Structured Merge Tree (TSM)

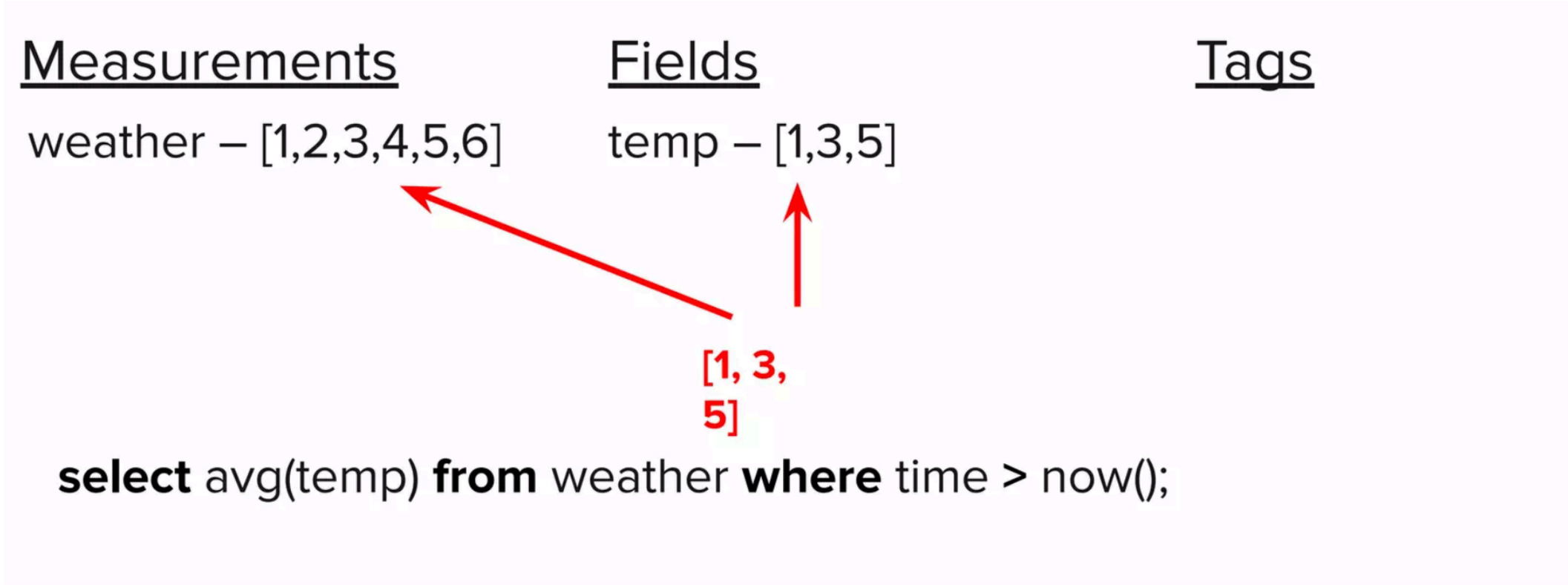
Posting List




```
select avg(temp) from weather where time > now() - 1h and state = “NY”;
```

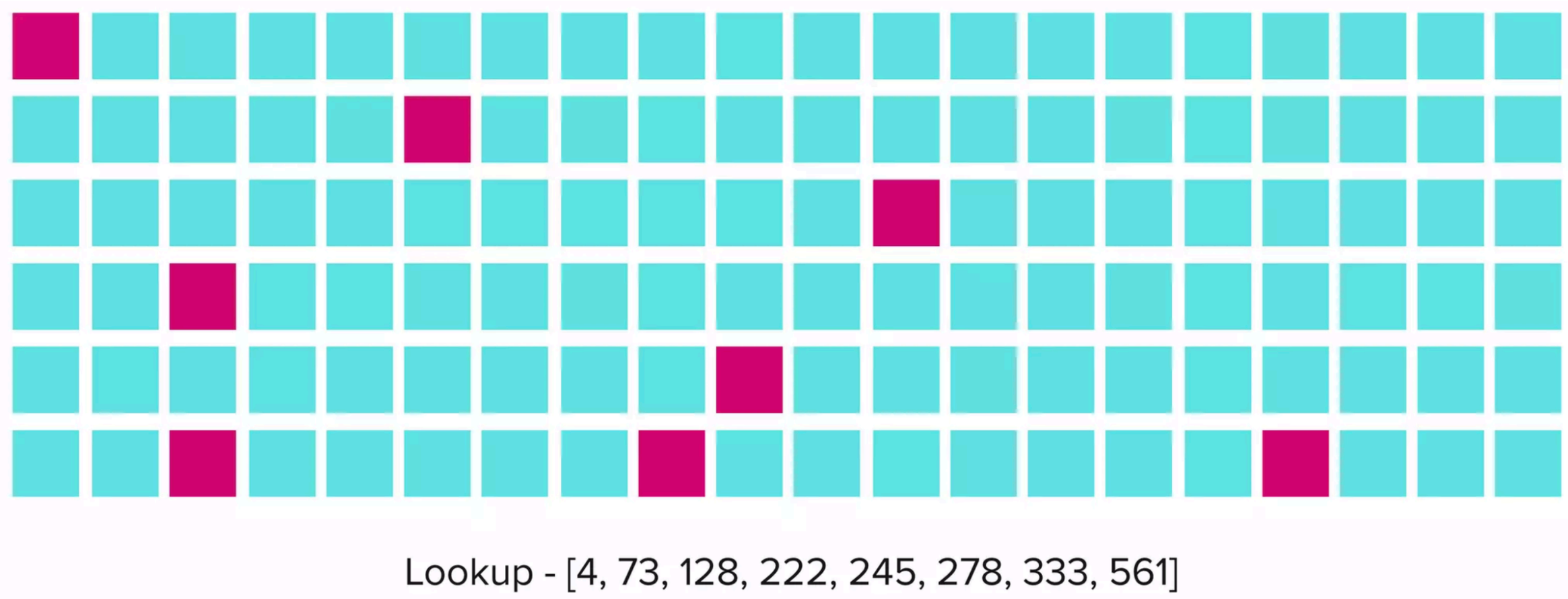
Time-Structured Merge Tree (TSM)

Posting List



Time-Structured Merge Tree (TSM)

Blocks



time asc order

Time-Structured Merge Tree (TSM)

Tradeoffs

- **Sehr schnell bei Lookups**
- **Nur schnell bei geringen Kardinalitäten**
- **Inverted Index muss gepflegt werden; Inserts**
- **Sortiert nach Zeit**

InfluxDB
!=
Big Data

Anforderungen

Verschiedene Typen von Daten

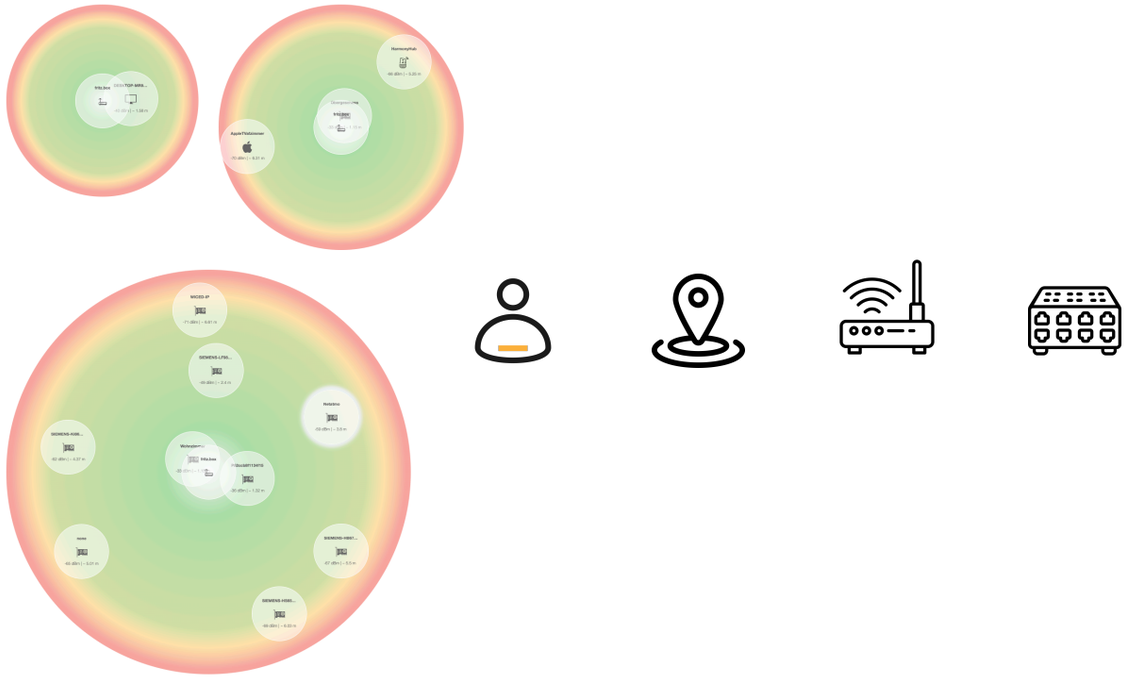
Relationale Daten Nicht Relationale Daten

Zeitbasierte Daten

Flexible Datenerfassung

TR-069 TR-369 SNMP

SQL-Statements

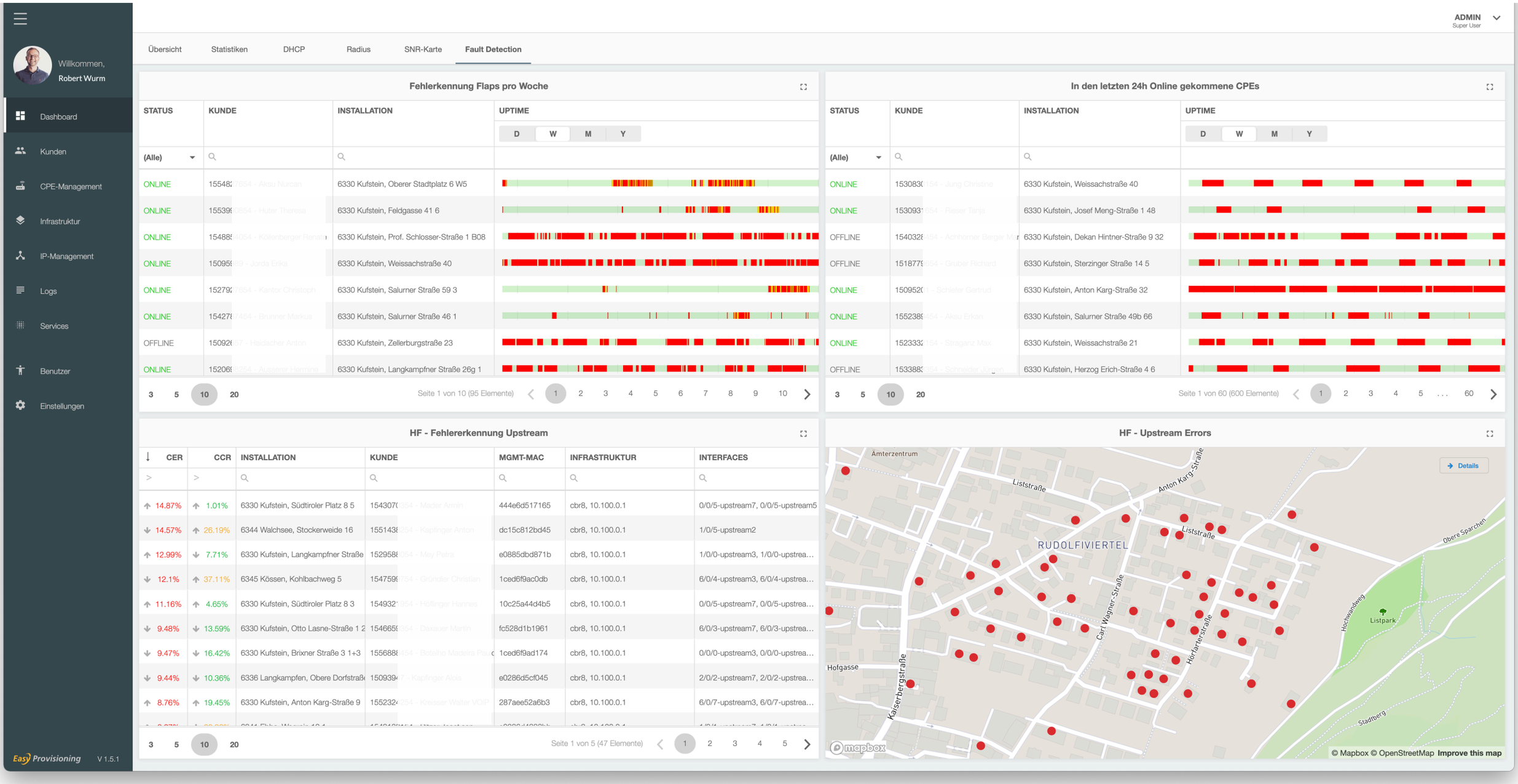


Effiziente Datenpersistierung

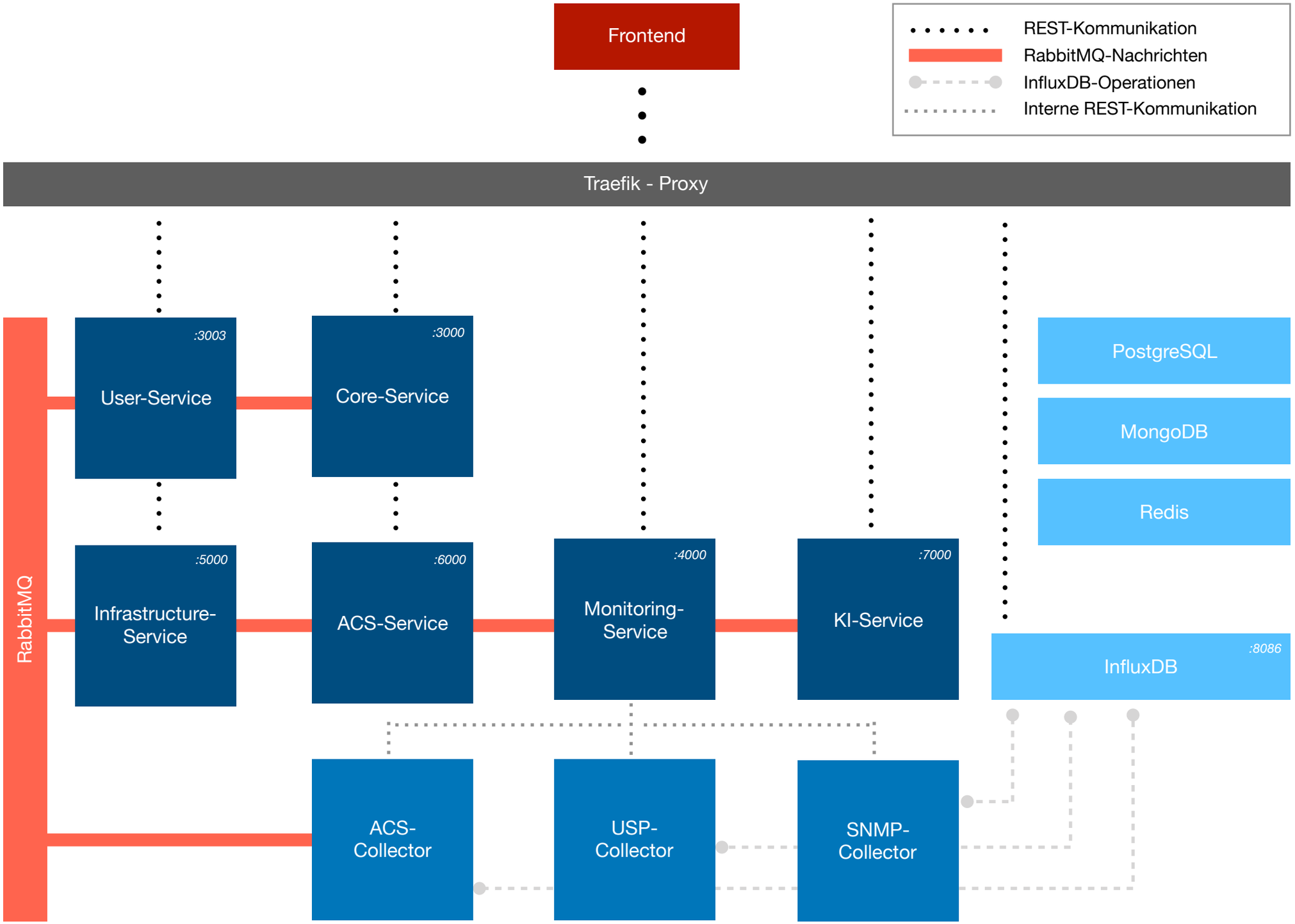
Speicher **Abfragezeiten**

Dynamik

Anforderungen



Microservice Architecture



https://github.com/wurmrobert/big_data_mit_InfluxDB

Fragen?

https://github.com/wurmrobert/MasterThesis

https://www.slideshare.net/influxdata/paul-dix-influxdata-the-journey-of-influxdb-influxdays-2022

https://www.slideshare.net/influxdata/paul-dix-influxdata-the-journey-of-influxdb-influxdays-2022

https://arxiv.org/pdf/2208.13982.pdf

https://docs.influxdata.com/influxdb/v1.8/concepts

https://www.youtube.com/watch?v=58nrv2SjZ-0

https://www.youtube.com/watch?v=sfHaYdcDaAY