

*Računarstvo i informatika*

*Katedra za računarstvo  
Elektronski fakultet u Nišu*

# Napredne baze podataka Vremenske serije podataka

Zimski semestar 2020/2021



# Vremenske serije podataka

- Promenljiva čije se vrednosti mere u određenim tačkama u vremenu se naziva **vremenskom serijom podataka (time series)**.
- Vremenski intervali između tačaka su često jednaki.
- Vremenske serije podataka imaju trostruku namenu:
  - **Razumevanje** pojava i struktura koje su proizvele određenu vremensku seriju podataka
  - **Nadgledanje (monitoring)** tih istih pojava i struktura
  - **Predviđanje** budućih vrednosti



# Vremenske serije podataka

- Široku praktičnu primenu vremenske serije podataka imaju u oblastima kao što su: ekonomска prognoza, vremenska prognoza, analiza budžeta, kontrola procesa i kvaliteta...
- U prirodi postoje razne veličine i pojave čije se vrednosti mogu meriti u vremenu u određenim vremenskim intervalima.
- Primeri: temperatura, količina padavina, prinos neke biljke...
- U računarstvu i informatici, vremenske serije podataka se često koriste za praćenje određenih performansi sistema, kako softvera, tako i hardvera.
- Beleže se i skladište podaci iz različitih izvora kako bi se stekla slika o performansama sistema i blagovremeno primetili i otklonili potencijalni problemi u radu.



# Vremenske serije podataka

- Takođe, primenjuju se i u oblastima kao što su mašinsko učenje i *data mining*.
- Sa napretkom različitih tehnologija gde je potrebno raditi sa vremenskim serijama, npr. „Internet Stvari“ (*IoT – Internet of Things*), količina podataka je sve veća i veća (*big data*) tako da primena vremenskih serija u računarstvu i informatici postaje sve veći izazov za koji je potrebno razvijati posebne alate i servise.



# Vremenske serije podataka

- **Definicija:** *Vremenska serija predstavlja niz vrednosti neke veličine, pribavljenih u sukcesivnim vremenskim intervalima, gde su intervali često jednaki.*
- Postoji četiri obrasca ili komponenti kojima se mogu opisivati promene vrednosti vremenske serije podataka, a to su:
  - Opšti trend (*general trend*) -  $f(t)$
  - Sezonska komponenta (*seasonal movements*) -  $s(t)$
  - Ciklična komponenta (*cyclical movements*) -  $c(t)$
  - Slučajna komponenta (*irregular fluctuations*) -  $\varepsilon(t)$



# Vremenske serije podataka

- **Opšti trend**

- Opštim trendovima se nazivaju dugoročne promene vrednosti vremenske serije podataka.
- Trend može pokazati tendenciju rasta ili pada neke vrednosti tokom dugačkog perioda.
- Trend ne mora biti linearan.
- Npr., socioekonomski faktori kao što su zaposlenost, visina plata itd. se obično menjaju u trendovima.

- **Sezonska komponenta**

- Sezonska komponenta predstavlja promene u nešto kraćim periodima koje se dešavaju usled određenih sezonskih uslova.
- Dužina sezone je uvek fiksna i poznata.
- Sezona, u ovom smislu, može biti: godišnji kvartal, mesec, nedelja itd.



# Vremenske serije podataka

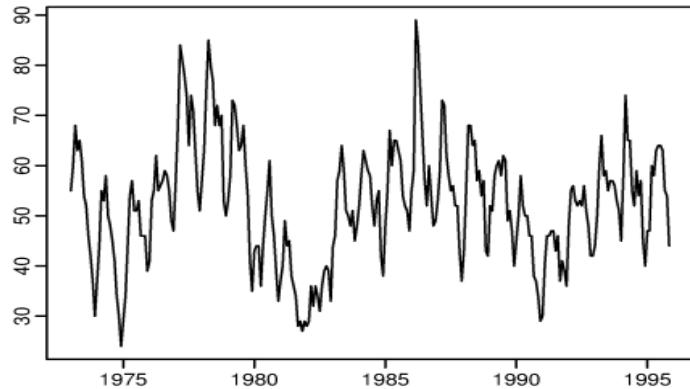
- **Ciklična komponenta**

- Ciklične komponente imaju podaci koji pokazuju rastove ili padove koji nisu u periodima fiksne dužine i po tome se razlikuju od sezonskih trendova.
- Da bi mogle biti proučavane, ovakve vremenske serije podataka obično moraju posedovati što više tačaka, sa što manje slučajnih komponenti.

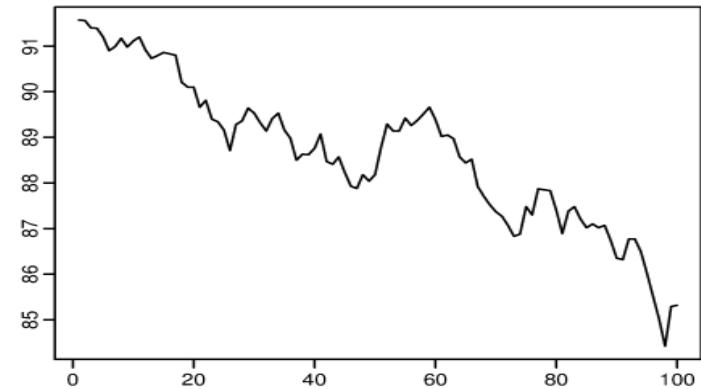
- **Slučajna komponenta**

- Slučajnu komponentu čine iznenadne promene koje se dešavaju u vremenskoj seriji koje imaju malu verovatnoću da se opet promene.
- Ovakve promene se ne mogu objasniti opštim trendovima, a ni sezonskim ili cikličnim promenama, takođe.
- Ovakve varijacije se nekada mogu nazivati iregularnim fluktuacijama.
- Te promene, iako su slučajne u prirodi, mogu nagovestiti kontinualnu promenu u trendovima u predstojećem periodu.

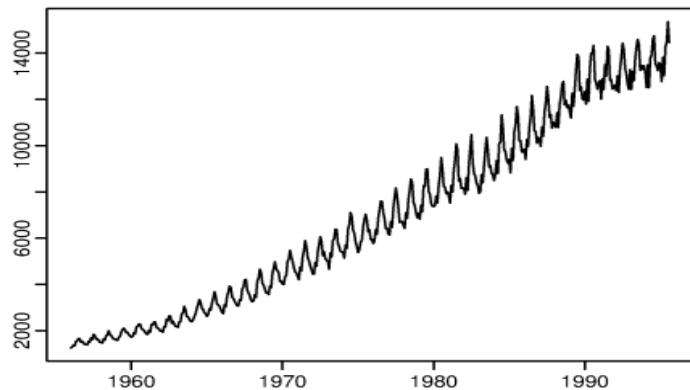
# Vremenske serije podataka



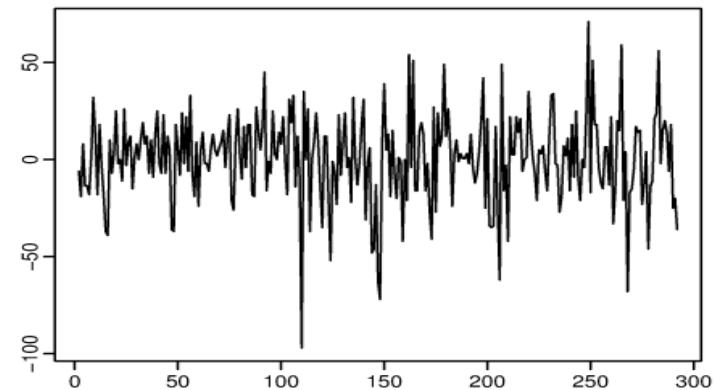
A) Broj prodatih nekretnina po mesecima



B) Vrednost akcija na berzi po danu



C) Mesečna proizvodnja struje u Australiji



D) Dnevne promene u Dow-Jones indeksu



# Vremenske serije podataka

- Na slici A se može videti jasno izražena sezonalnost, kao i određena cikličnost koja traje u periodima od 6-10 godina.
- Promena vremenske serije podataka na slici B) pokazuje jasno izražen negativni trend tj. opadajući generalni trend.
- Na slici C) se vidi trend rasta sa jako izraženom sezonalnošću.
- Dok se na slici D) ne mogu uočiti gotovo nikakvi trendovi ili sezonske i ciklične promene, već su tu promene uglavnom nasumične tj. ima dosta iregularnih fluktuacija, tako da nema jasno izraženih šabloni po kojima bi se mogao razviti model koji bi koristio za predviđanje vremenskih serija.



# Vremenske serije podataka

- Modeli vremenskih serija:
  - Aditivni model -  $X_t = f(t) + s(t) + c(t) + \varepsilon(t)$ 
    - Ovaj model prepostavlja da se sve komponente ponašaju nezavisno jedna od druge. Pa tako povećanje neke od komponenti ne izaziva povećanje ili smanjenje neke druge komponente. Aditivni model se koristi ako su sezonska, ciklična i slučajna komponenta nezavisne od kretanja trenda.
  - Multiplikativni model -  $X_t = f(t) \cdot s(t) \cdot c(t) \cdot \varepsilon(t)$ 
    - Ovaj model prepostavlja da sve komponente međusobno zavise jedna od druge. Pa tako, ako trend serije raste, fluktuacije izazvane sezonom, ciklusom ili slučajnim kretanjem će takođe rasti.
  - Mešoviti model -  $X_t = f(t) \cdot c(t) \cdot (s(t) + \varepsilon(t) - 1)$ 
    - Ovaj model predstavlja kombinaciju aditivnog i multiplikativnog modela. U njemu se prepostavlja da su sezonska i slučajna komponenta zavisne od trenda i ciklične komponente, ali da su one međusobno nezavisne.

# Time series databases

- **Time series databases (TSDB)** - baze podataka za rad sa vremenskim serijama podataka
- To su baze podataka koje su posebno optimizovane i namenjene za rad sa vremenskim serijama i često imaju ugrađene u sebe određene metode za analizu serija, kao i druge pogodnosti.
- Sama arhitektura različitih TSDB se razlikuje od implementacije do implementacije.
- Neke koriste NoSQL pristup, dok se druge baziraju na korišćenju tehnologije relacionih baza podataka u pozadini.
- Neke koriste već postojeće DBMS-ove i dodaju sloj funkcionalnosti za rad sa vremenskim serijama, dok druge koriste svoja rešenja i razvijaju i svoj upitni jezik (query language).

# Time series databases

- Postoje dva fundamentalna zahteva koje baza podataka rad sa vremenskim serijama podataka mora da zadovolji:
  - **Ko-lokacija podataka**
  - **Efikasan pristup podacima u proizvoljnom intervalu**
- Sekvencijalni pristup podacima je i dalje dosta brži od nasumičnog pristupa podacima, čak i sa SSD-ovima (solid-state drive).
- Sistem koji smešta sortirane podatke u sekvencijalnom redosledu (ko-lokacija) na istim mestima, što ide prirodno uz vremenske serije podataka, će učiniti te podatke dostupnim za brzo i efikasno čitanje.
- Forsiraju se CR- a ne CRUD operacije.

# Time series databases

- **Performanse i skalabilnost**

- Dobro projektovana TSDB omogućava lako skaliranje sistema tako da on može da podrži i milione tačaka u vremenskim serijama u kontinualnom toku podataka i da izvrši analize u realnom vremenu.

- **Kompakcija podataka (data compaction)**

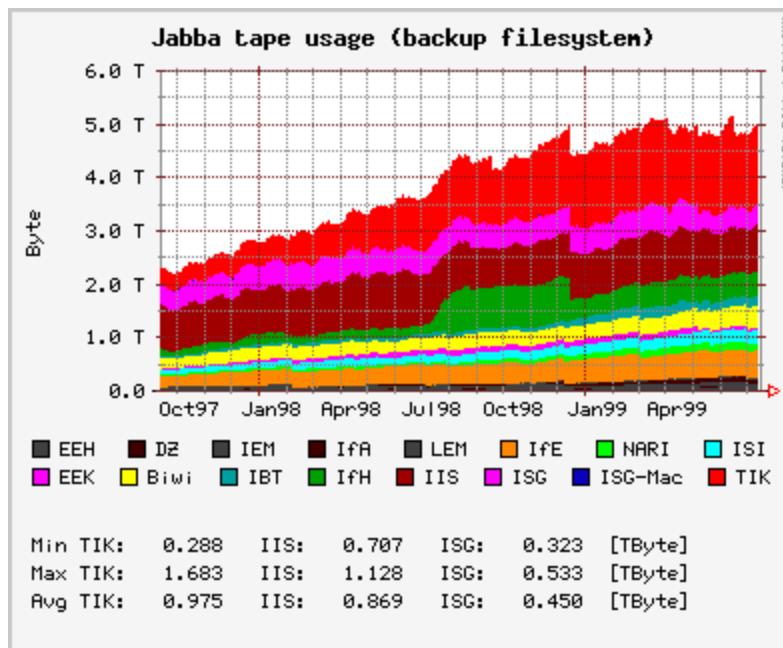
- Kako podaci neke vremenske serije podataka stare, to su manje bitne individualne tačke.
- Gotovo svaka TSDB ima određene mehanizme za degradiranje i združavanje podataka.
- To znači da se uglavnom visoko precizni podaci kratko čuvaju u bazi (npr. vrednost neke promenljive u intervalima od jedne sekunde), tim podacima se konstantno **smanjuje rezolucija (downsample)** i raznim matematičkim funkcija **agregacije (aggregate)** se objedinjuju u tačke sa većim periodima (npr. sada je vremenska serija aproksimirana sa tačkama između kojih je interval od jednog minuta)

# Time series databases

- **Niži troškovi**
  - Zbog optimizovanosti za rad sa vremenskim serijama, TSDB direktno utiču na smanjenje troškova zato što troše manje računarskih resursa pri radu sa vremenskim serijama podataka.
- **Bolje poslovne odluke**
  - Time što se podaci mogu analizirati u realnom vremenu, organizacije mogu da prave bolje i preciznije odluke, kao i da detektuju i reaguju na potencijalne probleme na vreme.
- Trentuno postoji veliki broj i komercijalnih i besplatnih varijanti baza podataka sa rad sa vremenskim serijama podataka koje se razlikuju po svojoj implementaciji i filozofiji.

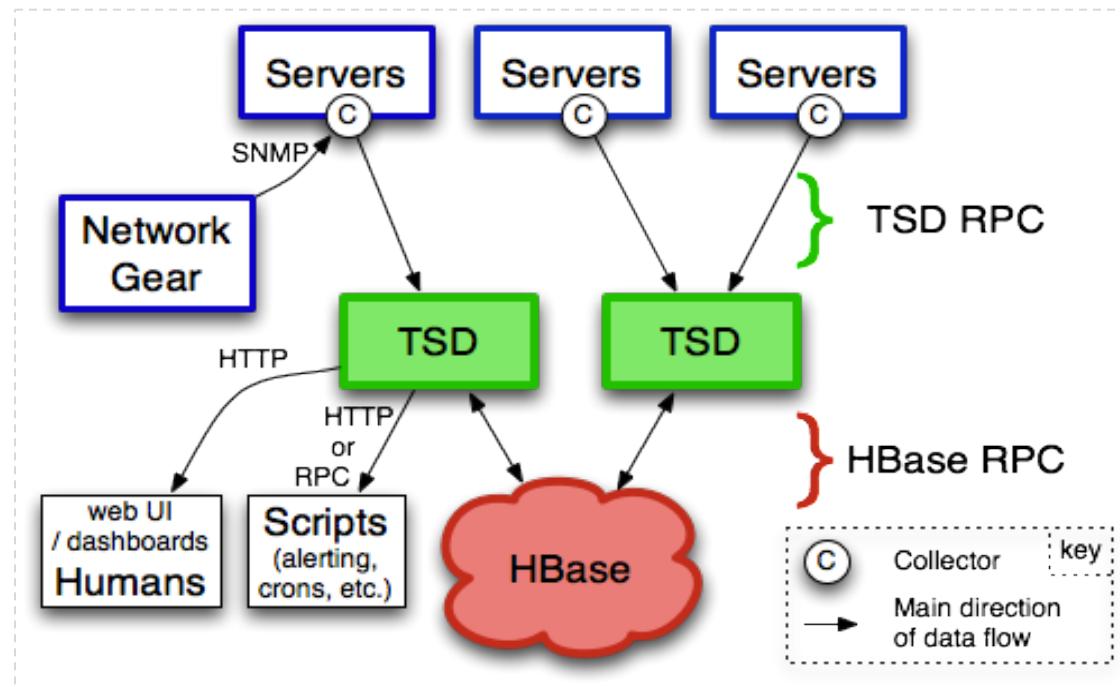
# RRDTool

- RRD Tool - Round-Robin Database tool  
<https://oss.oetiker.ch/rrdtool/index.en.html>
- RRDtool je napisan u programskom jeziku C.
- Podaci se smeštaju u bazi koja je implementirana kao kružni bafer.
- Prva verzija 1999 godine. Aktuelna verzija 2019.



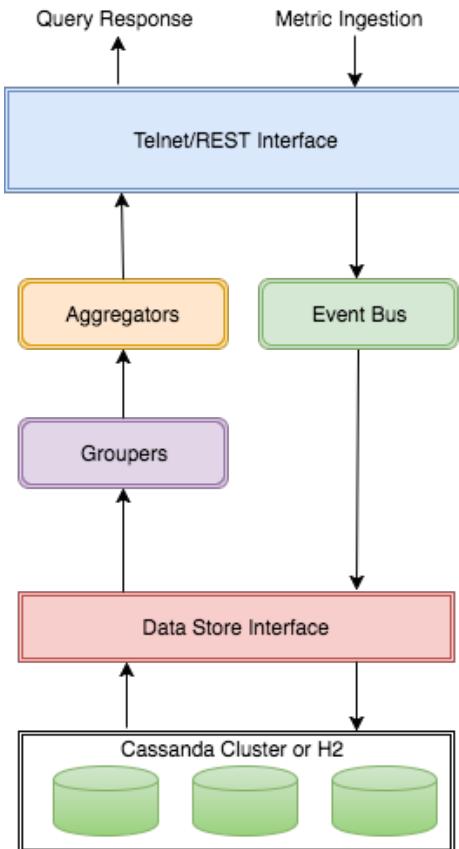
# OpenTSDB

- OpenTSDB (<http://opentsdb.net>) je napisan u Javi.
- OpenTSDB se sastoji od skupa Time Series Daemon-a (TSD).
- Za skladištenje podataka koristi Apache HBase bazu podataka ili Google-ovim Bigtable servis.



# KairosDB

- KairosDb ( <http://kairosdb.github.io/> ) je nastao kao fork OpenTSDB projekta.
- Za skladištenej podataka koristi Apache Cassandra ili H2 baze podataka.



# RiakTS

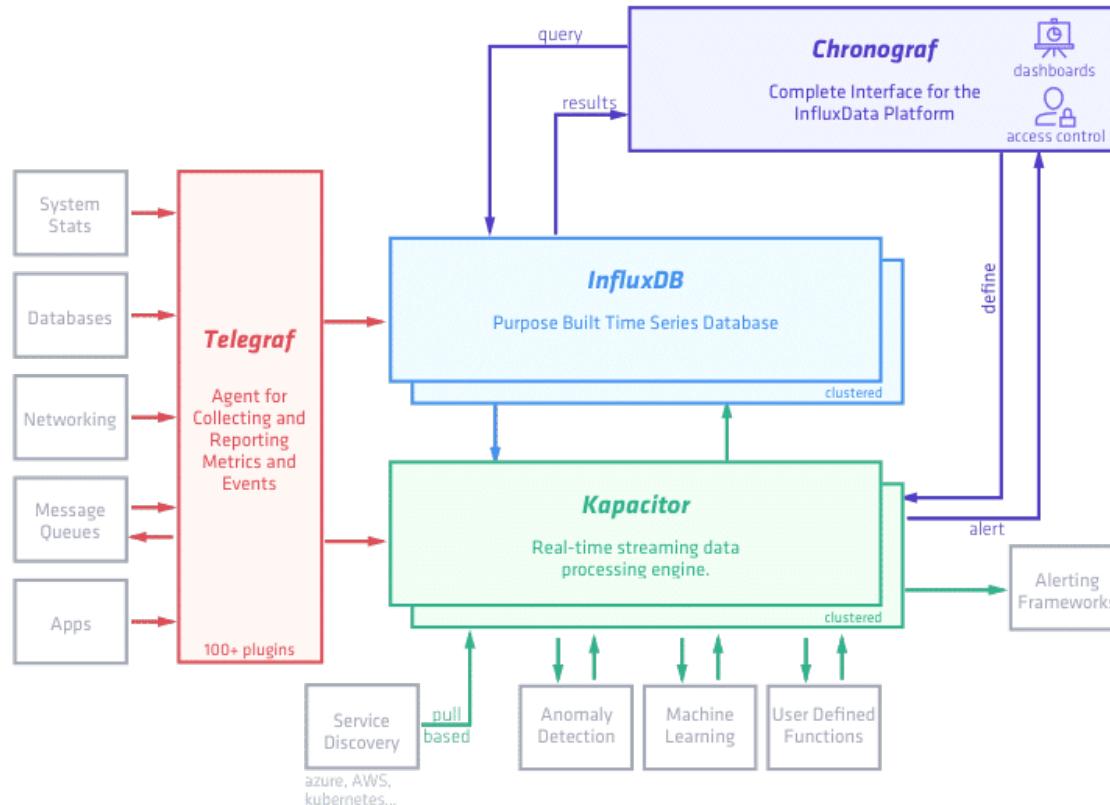
- RiakTS (<https://riak.com/products/riak-ts/index.html>) je distribuirana NoSQL key-value baza podataka optimizovana za skladištenje vremenskih serija podataka.
- Predstavlja nadogradnju postojećeg Riak key-value rešenja i dele isti kod.
- Implementiran korišćenjem Erlang programskog jezika.
- Postoji komercijalna i besplatna licenca.

# InfluxDB

- InfluxDB ( <https://www.influxdata.com/> )
- Implementiran korišćenjem jezika Go.
- Native TSDB rešenje.
- Postoji komercijalna i besplatna licenca.
- Deo TICK stack-a.
- InfluxDB ne bi trebalo da se koristi kao standardna CRUD (create, read, update, delete) baza podataka.
- Zbog prirode vremenskih serija podataka koje se gotovo nikad ne ažuriraju (update) i ne brišu (delete), InfluxDB je dosta optimizovaniji za čitanje i pisanje.
- Ne treba razmatrati primene sa intenzivnim ažuriranjem i brisanjem podataka zbog lošijih performansi.

# InfluxDB

- TICK stack



# InfluxDB

- **Telegraf**
  - Agent koji je zadužen za prikupljanje podataka iz različitih izvora (plug-in arhitektura).
- **Kapacitor**
  - Okruženje za obradu podataka (u realnom vremenu ili batch obradu).
- **Chronograph**
  - Korisnički interfejs za administraciju i vizualizaciju platforme.
- **InfluxDB**
  - Skladište za čuvanje vremenskih serija podataka

# InfluxDB

- Model podataka:
  - **Vreme (time)** je tip kolone i interno se pamti kao UNIX timestamp, a prikazuje se u RFC3339 [7] formatu
  - **Polja (fields)** se uglavnom koriste za smeštanje samih numeričkih vrednosti neke promenljive, mada nisu striktno vezana za brojeve. U poljima se mogu pamtitи integer-i, boolean-ovi, string-ovi ili float-i. Polja nisu indeksirana i treba izbegavati upite koji pretražuju po poljima zbog lošijih performansi.
  - **Tagovi (tags)** predstavljaju tip kolone koji se uglavnom koristi za neke propratne, meta-informacije koje mogu biti od značaja uz izmerene vrednosti vremenske serije podataka. Tagovi jesu indeksirani i preporuka je, ukoliko je potrebno, da se samo oni koriste pri upitima za pretragu.



# InfluxDB

- **Mera (measurement)** ili metrika se sastoji od minimum jedne kolone tipa vreme i jedne kolone tipa polje i kao takva, čini osnovnu strukturu podataka u InfluxDB-u. Mera može imati neograničen broj polja i tagova.

Vreme	Pozivi	Poruke	Lokacija
2017-09-01T00:00:00Z	12	56	Centar
2017-09-01T00:00:00Z	25	120	Duvanište
2017-09-02T00:00:00Z	4	12	Centar
2017-09-02T00:00:00Z	90	234	Palilua
2017-09-03T00:00:00Z	25	12	Centar
2017-09-03T00:00:00Z	80	48	Bulevar

# InfluxDB

- **InfluxQL** je naziv upitnog jezika koji je napravljen za potrebe InfluxDB-a i koristi se za pisanje upita.
- InfluxQL ne podržava spojeve.
- **Kontinualni upiti** su InfluxQL upiti koji se periodično izvršavaju nad podacima koji su dobijeni u realnom vremenu i čiji se rezultati pamte u zasebnim merama.
- Kontinualni upiti omogućavaju smanjenje rezolucije (downsampling) vremenskih serija podataka.

```
CREATE CONTINUOUS QUERY "cq_basic" ON "transportation"
```

```
BEGIN
```

```
    SELECT mean("passengers") INTO "average_passengers" FROM "bus_data" GROUP BY time(1h)
```

```
END
```

- Upit se izvršava svakog sata i izračunaće prosečan broj putnika (passengers) za taj sat, a potom upamtitи taj broj u meru average\_passengers.

# InfluxDB

- Politika zadržavanja (retention policy) – definiše kontinualno brisanje podataka određene starosti.
- Ako se, prilikom kreiranja baze, ne navede politika zadržavanja – ta baza će imati beskonačno zadržavanje odnosno podaci nikada neće biti obrisani.

```
CREATE RETENTION POLICY "one_day_only" ON "NOAA_water_database"  
DURATION 23h40m REPLICATION 1 DEFAULT
```

- Dužina zadržavanja može biti proizvoljna.
- Svi podaci u bazi koji su stariji od zadatog vremena (npr. 23 časova i 40 minuta) će biti konstantno brisani.

# InfluxDB

- Funkcije agregacije

Funkcija	Opis
<b>COUNT()</b>	Vraća ukupan broj tj. vrši prebrojavanje određenog polja ili taga iz mera.
<b>DISTINCT()</b>	Vraća spisak unikatnih vrednosti
<b>INTEGRAL()</b>	Vraća površinu ispod krive koja je iscrtana vrednostima polja koje zadovoljavaju uslov upita tj. računa integral te krive.
<b>MEAN()</b>	Vraća prosečnu vrednost od polja navedenih u upitu.
<b>MEDIAN()</b>	Vraća medijanu od polja navedenih u upitu.
<b>MODE()</b>	Vraća vrednost koja se najčešće javlja od polja/tagova navedenih u upitu.
<b>SPREAD()</b>	Vraća vrednost između maksimalne i minimalne vrednosti polja nevedenog u upitu.
<b>STDDEV()</b>	Vraća statističku standardnu devijaciju vrednosti polja nevedenog u upitu.
<b>SUM()</b>	Vraća ukupnu sumu svih vrednosti polja navedenog u upitu.

# InfluxDB

- Funkcije selekcije

Funkcija	Opis
<b>BOTTOM()</b>	Vraća N najmanjih vrednosti polja navedenog u upitu.
<b>FIRST()</b>	Vraća prvu vrednost polja/taga navedenog u upitu, najstariju po timestamp-u.
<b>LAST()</b>	Vraća poslednju vrednost polja/taga navedenog u upitu, najnoviju po timestamp-u.
<b>MAX()</b>	Vraća najveću vrednost polja navedenog u upitu.
<b>MIN()</b>	Vraća najmanju vrednost polja navedenog u upitu.
<b>PERCENTILE()</b>	Vraća N-ti postotak vrednosti polja navedenog u upitu
<b>SAMPLE()</b>	Vraća uzorak od N slučajnih vrednosti polja navedenog u upitu.
<b>TOP()</b>	Vraća N najvećih vrednosti polja navedenog u upitu..

# InfluxDB

- Funkcije za transformaciju

Funkcija	Opis
<b>CUMULATIVE_SUM()</b>	Vraća kumulativnu sumu vrednosti polja navedenog u upitu.
<b>DERIVATIVE()</b>	Vraća brzinu promene krive iscrtane vrednostima polja navedenog u upitu tj. prvi izvod krive.
<b>DIFFERENCE()</b>	Vraća rezultat oduzimanja susednih vrednosti polja navedenih u upitu.
<b>ELAPSED()</b>	Vraća proteklo vreme između susednih vrednosti polja/tagova navedenih u upitu.
<b>MOVING_AVERAGE()</b>	Vraća pokretni prosek od po N susednih vrednosti polja/tagova navedenih u upitu
<b>NON_NEGATIVE_DERIVATIVE()</b>	Vraća brzinu promene krive iscrtane vrednostima polja navedenog u upitu tj. prvi izvod krive samo za pozitivne vrednosti i vrednosti koje su jednake nuli.



# TimescaleDB

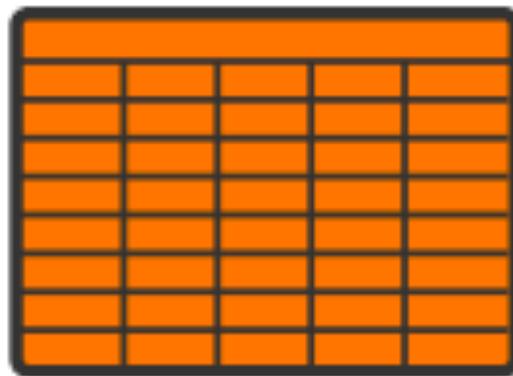
- TimescaleDB (<https://www.timescale.com/>)
- Open source rešenje
- TSDB rešenje koje je razvijeno kao ekstenzija PostgreSQL baze podataka
- TSDB rešenje koje razvijeno na osnovma relacione baze podataka.
- Nudi sve funkcionalnosti koje nudi i PostgreSQL baza podataka uz dodatak funkcionalnosti specifičnih za vremenske serije podataka.

# TimescaleDB

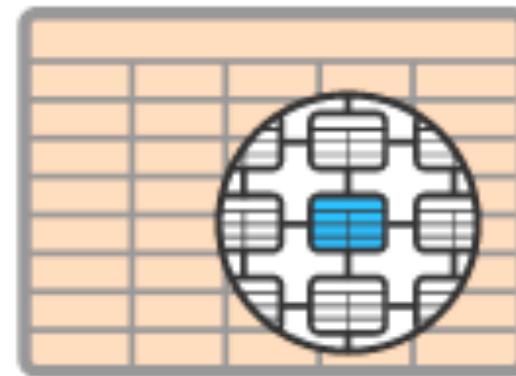
- Vremenske serije podataka se čuvaju u hipertabelama (hypertables).
- Hipertabela predstavlja virtualni pogled na veći broj individualnih tabela – chunks.
- Chunks nastaju particionisanjem hipertabele po jednoj ili više dimenzija.
- Svaka hipertabela je uvek particionisana po vremenu.
- Dodatno hipertabela može biti particionisana i po drugim dimenzijama: ID, lokacija, user ID, ...

# TimescaleDB

- Organizacija hipertabele



Hypertable



Chunk

# TimescaleDB

- Particije su implementirane korišćenjem standardnih tabele PostgreSQL baze podataka.
- Particije su međusobno disjunktne.
- Podaci su particonisani tako da veličina particija omogućava optimalne performanse prilikom operacija upisivanja podataka.
- Vremenom se vrši kompresija particija. Particija se iz formata gde su podaci organizovani po vrstama transformiše u oblik gde su podaci organizovani po kolonama.
- Kompresija je potpuno transparentna za korisnika.

# TimescaleDB

- Kreiranje hipertable

```
-- Do not forget to create timescaledb extension
CREATE EXTENSION timescaledb;

-- We start by creating a regular SQL table
CREATE TABLE conditions (
    time      TIMESTAMPTZ      NOT NULL,
    location  TEXT            NOT NULL,
    temperature DOUBLE PRECISION  NULL,
    humidity   DOUBLE PRECISION  NULL
);

-- Then we convert it into a hypertable that is partitioned by time
SELECT create_hypertable('conditions', 'time');
```

# TimescaleDB

- Rad sa vremenskim serijama podataka

```
INSERT INTO conditions(time, location, temperature, humidity)
VALUES (NOW(), 'office', 70.0, 50.0);

SELECT * FROM conditions ORDER BY time DESC LIMIT 100;

SELECT time_bucket('15 minutes', time) AS fifteen_min,
       location, COUNT(*),
       MAX(temperature) AS max_temp,
       MAX(humidity) AS max_hum
  FROM conditions
 WHERE time > NOW() - interval '3 hours'
 GROUP BY fifteen_min, location
 ORDER BY fifteen_min DESC, max_temp DESC;
```