PostgreSQL et TimescaleDB pour la collecte de métriques à haute fréquence

Simon Delamare

LIP / CNRS

5 décembre 2019

Données orientées *time series*

- Time series ?
  - données horodatées
  - temps + valeur + méta-données

time	value	device_id	metric_id
2019-11-28 08:16:54+00	77.186	nova-1	wattmetre_power_watt
2019-11-28 08:22:40+00	79.882	nova-1	wattmetre_power_watt
2019-11-28 08:32:49+00	74.966	nova-1	wattmetre_power_watt
2019-11-28 08:44:11+00	74.69	nova-2	wattmetre_power_watt
2019-11-28 08:49:52+00	77.146	nova-2	wattmetre_power_watt

Données orientées *time series*

- Time series ?
  - données horodatées
  - temps + valeur + méta-données

time	value	device_id	metric_id
2019-11-28 08:16:54+00	77.186	nova-1	wattmetre_power_watt
2019-11-28 08:22:40+00	79.882	nova-1	wattmetre_power_watt
2019-11-28 08:32:49+00	74.966	nova-1	wattmetre_power_watt
2019-11-28 08:44:11+00	74.69	nova-2	wattmetre_power_watt
2019-11-28 08:49:52+00	77.146	nova-2	wattmetre_power_watt

- (nombreux) exemples : monitoring des systèmes informatiques, IoT, enregistrements d'évènements, ...
- Bases de données orientées *time series* :
  - InfluxDB, OpenTSDB, KairosDB (NoSQL)
  - **TimescaleDB (extension PostgreSQL)**

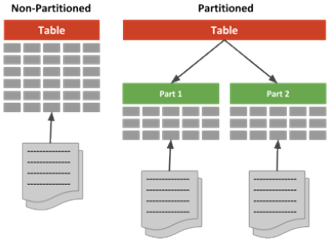
TimescaleDB

Pourquoi PostgreSQL ?

- SQL : syntaxe, cohabitation avec des données classiques
- Écosystème : clients, outils, savoir-faire, communauté, robustesse

Mais... pas optimisé pour les *time series*

- Beaucoup de données $\Longrightarrow$ Grosses tables
  $\Longrightarrow$ Pas en RAM (les indexes) $\Longrightarrow$ Faibles perfs
- Solution : partitionnement (= on découpe les tables en sous-tables).
  Mais doit être réalisé par l'admin de la DB



(source: oracle-base.com)

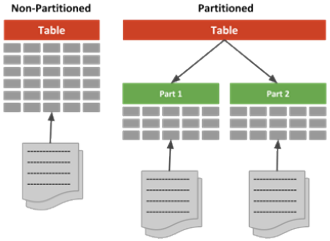
TimescaleDB

Pourquoi PostgreSQL ?

- SQL : syntaxe, cohabitation avec des données classiques
- Écosystème : clients, outils, savoir-faire, communauté, robustesse

Mais... pas optimisé pour les *time series*

- Beaucoup de données $\Longrightarrow$ Grosses tables
  $\Longrightarrow$ Pas en RAM (les indexes) $\Longrightarrow$ Faibles perfs
- Solution : partitionnement (= on découpe les tables en sous-tables).
  Mais doit être réalisé par l'admin de la DB



(source: oracle-base.com)

TimescaleDB

- Auto partitionnement sur des périodes de temps
- Les données insérées / accédées le sont généralement sur la partition la plus «récente»
- Cette partition est suffisamment petite pour tenir en RAM $\Longrightarrow$ Performances++
- Également : compression, agrégation de données, nouvelles opérations temporelles, ...

Monitoring à haute fréquence avec Kwapi (NG)

Prototype, déployé à l'essai dans Grid'5000

Collecte et mise à disposition de métriques (données de capteur, mesures de perf.) :

- Wattmetre haute fréquence : 50 mesures par seconde
- SNMP, IPMI : Capteurs sur les interfaces de management des machines, équipements réseaux, PDUs...
- Focus sur les données «externes» aux machines

jres DIJON 2019

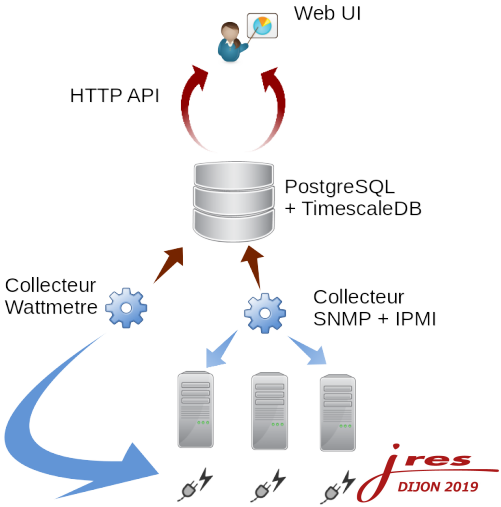
Monitoring à haute fréquence avec Kwapi (NG)

Prototype, déployé à l'essai dans Grid'5000

Collecte et mise à disposition de métriques (données de capteur, mesures de perf.) :

- Wattmetre haute fréquence : 50 mesures par seconde
- SNMP, IPMI : Capteurs sur les interfaces de management des machines, équipements réseaux, PDUs...
- Focus sur les données «externes» aux machines

Architecture :



Premier retour d'expérience

- Déployé sur une VM «modeste» : 4 vCPUs @2.3Ghz, 16Go RAM
- Données collectées : $> 3.10^6$ entrées/h
  - 30 valeurs/sec via SNMP (=10 requêtes SNMP/sec)
  - 1000 valeurs/sec via wattmetre (16 nœuds à haute fréquence, 120 nœuds à 1Hz)
  - 1Go de données par heure, 20Mo compressé
- Requêtes :
  - $\approx$ 1 sec. pour 10^5 lignes (ex: toutes les métriques d'un cluster de machine pour les 4 dernières heures)
  - $\approx$ 7 sec. pour 10^6 lignes (ex: toutes les métriques des wattmetres à haute fréquence pour la dernière heure)
  - $\approx$ 7 sec. pour 10^5 lignes qui ne sont pas toutes en RAM (ex: toutes les métriques d'une machine depuis 1 jour)
- Quelques bugs...

jres DIJON 2019

Premier retour d'expérience

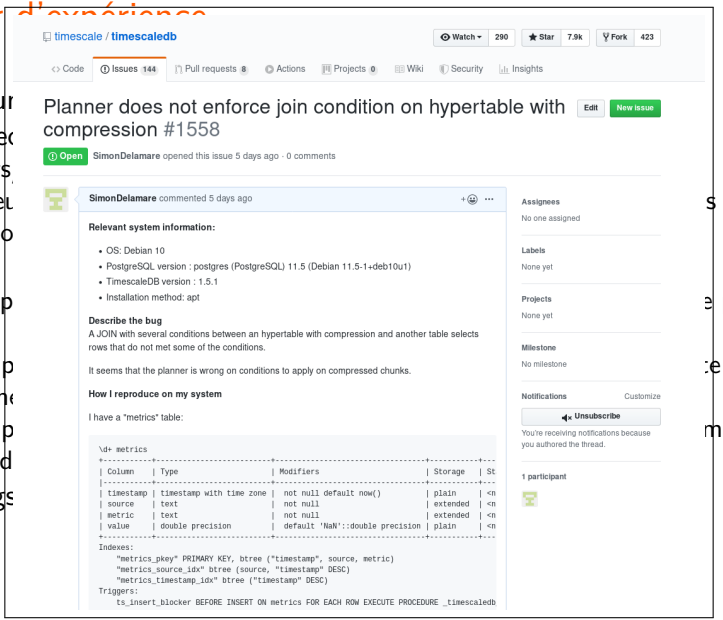
- Déployé sur ur
- Données collec
  - 30 valeurs
  - 1000 valeu
  - 1Go de do
- Requêtes :
  - ≈ 1 sec. p
    heures)
  - ≈ 7 sec. p
    dernière h
  - ≈ 7 sec. p
    machine d
- Quelques bugs

s à 1Hz)

e pour les 4 dernières

e fréquence pour la

métriques d'une



Conclusion