Σύγκριση μεταξύ TimescaleDB και InfluxDB

Λαουρεντιάν Γκούμε (A.M. 03118014)
Μαρία Τσιγάρα (A.M. 03118823)
Κρις Κούτση (A.M. 03118905)

I. Εισαγωγη

Η παρούσα εργασία περιλαμβάνει την εγκατάσταση και ρύθμιση των InfluxDB και TimescaleDB, τη δημιουργία και τη φόρτωση ενός μεγάλου όγκου δεδομένων σε κάθε σύστημα, τη δημιουργία επερωτημάτων (queries) με σκοπό τον έλεγχο της απόδοσης, και τη σύγκριση μετρικών απόδοσης για τα δύο συστήματα. Μάλιστα, η συγκριτική ανάλυση ανάμεσα στα δύο δημοφιλή αυτά συστήματα βάσεων δεδομένων αποτελεί ουσιαστικά και τον σκοπό της εργασίας μας, καθώς μέσω αυτής της ανασκόπησης θα μπορέσουμε να εμβαθύνουμε και τελικώς να κατανοήσουμε το τι πραγματικά δύναται να προσφέρει το κάθε ένα.

II. Εγκατασταση & Σεταρισμα

A. Υποδομή

Η υποδομή μας αποτελείται από ένα Ubuntu (16.04 LTS) Virtual Machine, το οποίο διαχειριζόμαστε μέσω του Okeanos που μας παρέχει και τους απαραίτητους πόρους. Το μηχάνημα αυτό διαθέτει 4 πυρήνες CPU, 8GB κύριας μνήμης, καθώς και μία δημόσια IP για πρόσβαση στο ίντερνετ.

B. Σετάρισμα Βάσεων

Στη συνέχεια, στήσαμε την TimescaleDB, ακολουθώντας τα βήματα του επίσημου οδηγού [1]. Αντίστοιχα, στήσαμε και την InfluxDB, βασιζόμενοι στο documentation [2]. Όσον αφορά την InfluxDB σημειώνουμε ότι:

- Δεν διαθέτει δωρεάν clustered έκδοση.
- Η InfluxDB 2.X δεν υποστηρίζεται από το TSBS [3].

Όσον αφορά την TimescaleDB σημειώνουμε ότι παρόλο που διατίθεται clustered έκδοση επιλέγουμε να μην την υιοθετήσουμε για τους κάτωθι λόγους [4]:

- Η υποστήριξη της multi-node έκδοσης έχει καταργηθεί από την έκδοση 2.13 και μετά.
- Όπως έχει αποδειχθεί από τη χαμηλή υιοθέτηση της, δεν είναι κοινώς αποδεκτό ότι είναι επωφελής.
- Απεναντίας, η single-node έκδοση έχει δεχθεί σημαντικές βελτιώσεις στην απόδοση εγγραφής και ανάγνωσης (10x για συνήθη queries).

Συμπερασματικά, αποφανθήκαμε ότι η single-node έκδοση δύναται, επί της παρούσης, να προσφέρει στην εργασία μας περισσότερα απ' ότι η multi-node έκδοση. Πέραν αυτού, δεδομένου ότι το ένα εκ των δύο συστημάτων δε διαθέτει clustered έκδοση, η όποια μεταξύ τους σύγκριση έχει νόημα μόνο σε αντίστοιχη, single node χρήση.

C. Σετάρισμα σουίτας benchmarking

Έπειτα, αξιοποιήσαμε την σουίτα του TSBS [5]. Το Time Series Benchmark Suite (TSBS) είναι ένα σύνολο προγραμμάτων που έχουν γραφεί στη γλώσσα προγραμματισμού Go και χρησιμοποιούνται για να δημιουργήσουν σύνολα δεδομένων και να διεξαγάγουν δοκιμές απόδοσης, ώστε να καταγραφεί ουσιαστικά το πόσο γρήγορα μπορούν οι βάσεις μας να διαβάσουν και να γράψουν τα ζητούμενα στην εκάστοτε περίπτωση δεδομένα. Σκοπός του είναι να μπορεί να "επεκτείνεται", ώστε να βρίσκουν σε αυτό αντίκρισμα ποικιλόμορφα use cases, queries και βάσεις, με αποτέλεσμα ο κάθε ενδιαφερόμενος διαχειριστής να μπορεί να βρει την πιο κατάλληλη γι' αυτόν. Προαπαιτούμενο για τη λειτουργία της σουίτας είναι η Go, την οποία και στήσαμε ορίζοντας κατάλληλα τις απαιτούμενες μεταβλητές περιβάλλοντος. Έχοντας ολοκληρώσει επιτυχώς τα παραπάνω, το tsbs repo που κάναμε clone, αντιστοιχεί στη θέση project1 της παρακάτω εικόνας:

```
/usr/local/
└── go/                 <-- This is your GOROOT
    ├── bin/            <-- Go binaries (go, gofmt, etc.)
    ├── src/            <-- Go source files (standard library)
    ├── pkg/            <-- Compiled package archives
    └── ...

/home/user/
└── go/                 <-- This is your GOPATH
    ├── bin/            <-- Compiled binaries of your projects
    ├── pkg/            <-- Compiled packages of your projects
    └── src/            <-- Source code of your projects
        ├── project1/   <-- Each project has its own directory
        │   ├── go.mod  <-- go.mod file for project1
        │   └── ...     <-- Other files and directories for project1
        └── project2/
            ├── go.mod
            └── ...
```

III. Παραγωγη Δεδομενων

Επόμενο βήμα μας ήταν η παραγωγή δεδομένων για αρχεία μικρού, μεσαίου και μεγάλου μεγέθους, τόσο για τη βάση InfluxDB, όσο και για την TimescaleDB (συνολικά 6 βάσεις / 6 datasets). Ενδεικτικά[1] παρουσιάζουμε την εντολή που τρέξαμε για την βάση InfluxDB για το μεγάλο αρχείο:

```
tsbs_generate_data --use-case="devops" --seed=123 \
--scale=10 --timestamp-start="2016-01-01T00:00:00Z" \
--timestamp-end="2016-02-12T00:00:00Z" --log-interval="10s" \
--format="influx" | gzip > ./datasets/influx_big.gz
```

[1] Το σύνολο των εντολών θα φορτωθεί για διευκόλυνση σε bash script, για να εκτελεστεί.

Αποσαφηνίζουμε τα παραπάνω flags ως ακολούθως :

- use case: devops, cpu ή iot
- seed: κλειδί (αρχικοποίησης) για τον αλγόριθμο PRNG / εξασφαλίζει την επαναληψιμότητα της διαδικασίας δημιουργίας τυχαίων αριθμών, δεδομένου ότι με ίδιο seed προκύπτει η ίδια ακριβώς ακολουθία τυχαίων τιμών
- scale: αριθμός συσκευών προς δημιουργία
- timestamp-start: ώρα έναρξης
- timestamp-end: ώρα λήξης
- interval: πόσος χρόνος πρέπει να περάσει μεταξύ κάθε ανάγνωσης ανά συσκευή
- format: InfluxDB / TimescaleDB

Ακόμη, επί της παραγωγής των δεδομένων, σημειώνουμε ότι παρήχθησαν σε zip αρχεία, για να μην ξεπεράσουν το χώρο του υπολογιστή (30 GB), καθώς το gzip δημιούργησε περίπου υποδεκαπλάσια σε μέγεθος datasets. Επιπλέον, τα μεγέθη των αρχείων ορίστηκαν ως κάτωθεν, για να πληρούν τις ζητούμενες προϋποθέσεις:

- αρχείο big: τουλάχιστον 8 GB (μεγαλύτερο από την κύρια μνήμη)
- αρχείο medium: λίγα GBs (~2-3GB)
- αρχείο small: κάποιες εκατοντάδες MBs (~300-500 MB)

Αξίζει να σημειώσουμε ότι ενώ φτιάχναμε τα datasets για τις InfluxDB / TimescaleDB, παρόλο που είχαν τα ίδια arguments αναφορικά με τα start / end timestamps, τα intervals, αλλά και το scale, τα αρχεία είχαν σημαντικές διαφοροποιήσεις ως προς το τελικό τους μέγεθος. Ειδικότερα, η διαφορά στο μέγεθος των αρχείων οφείλεται στη χρήση διαφορετικών μορφών σειριοποίησης δεδομένων από την InfluxDB και την TimescaleDB. Όταν η συνάρτηση tsbs_generate_data δημιουργεί δεδομένα, τα σειριοποιεί σε μορφή κατάλληλη για τη συγκεκριμένη βάση. Στην προκειμένη, για την InfluxDB χρησιμοποιεί το line protocol της, ενώ για την TimescaleDB χρησιμοποιεί ένα format παρόμοιο με αυτό του CSV. Ωστόσο, το line protocol της InfluxDB περιλαμβάνει το όνομα της μέτρησης, το σύνολο των ετικετών, το σύνολο πεδίων και τη χρονική σήμανση για κάθε εγγραφή, ενώ τουναντίον το format της TimescaleDB, οργανώνει τα δεδομένα σε γραμμές και στήλες, με τα ονόματα των στηλών να μην επαναλαμβάνονται για κάθε σημείο των δεδομένων. Έτσι, ακόμα κι αν χρησιμοποιούνται τα ίδια δεδομένα μέτρησης, ο τρόπος σειριοποίησης κατά τη δημιουργία διαφοροποιεί τα μεγέθη των αρχείων. Συνεπώς, ουσιαστικά βλέπουμε ότι οι βάσεις της TimescaleDB έχουν έναν έξτρα πίνακα tags, ο οποίος χρησιμοποιείται ως foreign key για τα relations μεταξύ των άλλων πινάκων. Αντιθέτως, στην InfluxDB δεν υπάρχει κάτι αντίστοιχο, οπότε όλα τα metadata αναπαράγονται σε κάθε πίνακα πολλαπλές φορές. Με αυτόν τον τρόπο, η TimescaleDB επιτυγχάνει πολύ μικρότερους χώρους στα αρχεία της, με ποσοστό που κυμαίνεται γύρω στο 30-35%.

IV. ΦΟΡΤΩΣΗ ΔΕΔΟΜΕΝΩΝ

Προχωρήσαμε στη φόρτωση των δεδομένων χρησιμοποιώντας τα εκτελέσιμα που μας παρέχει η σουίτα. Για παράδειγμα, για να φορτώσουμε τα δεδομένα στη big timescale βάση:

```
cat ./datasets/timescale_big.gz | gunzip | tsbs_load_timescaledb \
--host="localhost" --port=5432 --pass="12345678" --user="postgres" \
--workers=4 --do-create-db=false --do-abort-on-exist=false --db-name="big"
```

Διευκρινίζουμε ότι στην παραπάνω εντολή το μόνο flag που στην πραγματικότητα μεταβάλλουμε είναι ο αριθμός των workers, που δηλώνει το πλήθος των διεργασιών που χρησιμοποιούμε για τη φόρτωση των δεδομένων. Ωστόσο, προσαρμόζουμε το password μας (12345678) και το db-name που ορίστηκε ως big. Οι υπόλοιπες παράμετροι, παραδείγματος χάριν τα chunk-time και το field-index-count, είναι ορισμένες από default και οι τιμές τους παραμένουν σταθερές.

V. ΔΗΜΙΟΥΡΓΙΑ QUERIES

Αφού κάναμε το load, προκειμένου να εξετάσουμε τα αποτελέσματα που προέκυψαν, δημιουργούμε τα queries. Αυτό πραγματοποιείται με τη συνάρτηση tsbs_generate_queries, όπου τα flags αντιστοιχούν στα flags από το data generation που αναλύθηκαν στην αντίστοιχη παράγραφο. Σημειώνουμε ότι και οι τιμές τους ταυτίζονται, με εξαίρεση το timestamp_end που τίθεται κατά ένα δευτερόλεπτο πιο μετά, ενώ αποφασίσαμε να διατηρήσουμε σταθερό αριθμό queries (1000) για κάθε query type. Επίσης, τα συμπιέζουμε, για να πιάσουν λιγότερο χώρο στον δίσκο. Συμπληρωματικά, αποσαφηνίζουμε τα επιπλέον flags:

- queries: πλήθος queries
- query-type: τύπος query

Παρατίθεται ενδεικτικά και η εντολή για το small TimescaleDB single-group-1-1-1 query:

```
tsbs_generate_queries --use-case="devops" --seed=123 \
--scale=10 --timestamp-start="2016-01-01T00:00:00Z" \
--timestamp-end="2016-01-02T13:45:01Z" --queries=1000 \
--query-type="single-groupby-1-1-1" --format="timescaledb" \
| gzip > "/timescale_small/timescaledb_single_groupby-1-1-1.gz"
```

VI. ΜΕΤΡΗΣΗ ΔΕΙΚΤΩΝ ΑΠΟΔΟΣΗΣ

A. Στατιστικά εκτέλεσης queries και φόρτωσης δεδομένων

Έχοντας φορτώσει τα δεδομένα κι αφότου δημιουργήσαμε τα queries, όπως περιεγράφηκε στην προηγούμενη ενότητα, εκτελούμε το script tsbs_run_queries, για να τρέξουμε τα queries, να μετρήσουμε την απόδοση των ερωτημάτων αυτών και τελικώς να μπορέσουμε να προβούμε στην επιθυμητή σύγκριση των βάσεών μας. Αναφορικά με τα flags στο σημείο αυτό, αξίζει μόνο να σημειώσουμε ότι στην προκειμένη ως “workers” ορίζουμε το πλήθος των διεργασιών στις οποίες κατανέμουμε την εκτέλεση των queries. Παρατίθενται ενδεικτικά και οι εντολές για την εκτέλεση του single-groupby-1-1-1 στο small timescaledb και στο small influxdb με 4 και 2 workers αντίστοιχα:

```
/usr/bin/time -v sh -c "cat /home/user/go/src/tsbs/queries/timescale_small/timescaledb_single-groupby-1-1-1.gz \
| gunzip | /home/user/go/bin/tsbs_run_queries_timescaledb --workers=4 \
--postgres=\"host=localhost password=12345678 user=postgres sslmode=disable database=small\""
```

```
/usr/bin/time -v sh -c "cat /home/user/go/src/tsbs/queries/influx_small/influx_single-groupby-1-1-1.gz \
| gunzip | /home/user/go/bin/tsbs_run_queries_influx --workers=2 --db-name=small"
```

Σε αυτό το σημείο, αξίζει να εμβαθύνουμε στο τι εκφράζουν οι μετρικές που αξιοποιούμε για τη μέτρηση της απόδοσης. Πιο αναλυτικά, έχουμε:

- Wall clock time: Αναφέρεται στον πραγματικό χρόνο που περνάει από την αρχή μέχρι το τέλος μιας διεργασίας, συμπεριλαμβάνοντας όλους τους χρόνους αναμονής και είναι ουσιαστικά ο χρόνος που αντιλαμβανόμαστε ως χρήστες.
- Queries/second: Ο μέσος όρος των εκτελεσμένων επερωτημάτων ανά μονάδα χρόνου, που μας δίνει μια εικόνα της ταχύτητας απόκρισης της βάσης δεδομένων.
- User time: Ο χρόνος CPU που αφιερώνεται στην εκτέλεση των εργασιών στο user space κι αντιπροσωπεύει τον χρόνο που η CPU περνά εκτελώντας τον κώδικα της εφαρμογής.
- System time: Ο χρόνος CPU που αφιερώνεται σε συστημικές εργασίες, όπως είναι οι κλήσεις συστήματος.
- CPU Usage: Το ποσοστό της CPU που χρησιμοποιείται κατά τη διάρκεια εκτέλεσης μιας διεργασίας. Υψηλά ποσοστά CPU usage υποδηλώνουν έντονη δραστηριότητα.
- Total time: Ο συνολικός χρόνος που απαιτείται για τη φόρτωση των δεδομένων στη βάση.
- Rows/second: Ο μέσος όρος των εισαχθέντων σειρών δεδομένων ανά δευτερόλεπτο.
- Metrics/second: Ο μέσος όρος των μετρήσεων που εισάγονται στη βάση δεδομένων ανά δευτερόλεπτο.
- Data compression: Ο αποθηκευτικός χώρος που καταλαμβάνει η εκάστοτε βάση συγκριτικά με τον αποθηκευτικό χώρο που καταλαμβάνει το αντίστοιχο ασυμπίεστο dataset, ήτοι, το ποσοστό συμπίεσης.

Δεδομένου, λοιπόν, ότι οι μετρικές απόδοσης παρέχουν σημαντικές πληροφορίες για τα συστήματα των βάσεων δεδομένων που μελετάμε και λαμβάνοντας υπόψη την ανάλυση που προηγήθηκε, υπογραμμίζουμε ότι σε ορισμένες μετρικές, επιδιώκουμε να έχουν χαμηλές τιμές, ενώ σε άλλες επιθυμούμε τις μεγαλύτερες δυνατές. Από όσες αναλύσαμε, το να έχουμε ως αποτέλεσμα, όσο μικρότερη τιμή τόσο το καλύτερο, εμπίπτει στις: wall clock time, user time, system time, CPU usage και total_time, ενώ πλεονεκτούν οι μεγαλύτερες στις: queries/sec, rows/sec και metrics/sec. Όσον αφορά το data compression, πλεονεκτεί η βάση που αναλογικά καταλαμβάνει τον λιγότερο αποθηκευτικό χώρο, δηλαδή, η βάση που έχει χαμηλότερο τον λόγο “μέγεθος βάσης”/”μέγεθος dataset”. Η βάση αυτή, θα έχει το μεγαλύτερο ποσοστό συμπίεσης.

Υποσημειώνουμε τα queries που χρησιμοποιήθηκαν, καθώς και οι αντίστοιχες περιγραφές τους:

- single-groupby-1-1-1: Η μέγιστη τιμή για 1 μετρική για 1 host, κάθε πέντε λεπτά για 1 ώρα.
- single-groupby-1-1-12: Η μέγιστη τιμή για 1 μετρική για 1 host, κάθε πέντε λεπτά για 12 ώρες.
- single-groupby-1-8-1: Η μέγιστη τιμή για 1 μετρική για 8 hosts, κάθε πέντε λεπτά για 1 ώρα.
- single-groupby-5-1-1: Η μέγιστη τιμή για 5 μετρικές για 1 host, κάθε πέντε λεπτά για 1 ώρα.
- single-groupby-5-1-12: Η μέγιστη τιμή για 5 μετρικές για 1 host, κάθε πέντε λεπτά για 12 ώρες.
- single-groupby-5-8-1: Η μέγιστη τιμή για 5 μετρικές για 8 hosts, κάθε πέντε λεπτά για 1 ώρα.
- cpu-max-all-1: Aggregate όλων των μετρικών CPU ανά ώρα, για 1 host.
- cpu-max-all-8: Aggregate όλων των μετρικών CPU ανά ώρα, για 8 hosts.
- double-groupby-1: Aggregate τόσο στον χρόνο όσο και στον host, επιστρέφοντας τον μέσο όρο 1 μετρικής CPU ανά host, ανά ώρα, για μια μέρα.
- double-groupby-5: Aggregate τόσο στον χρόνο όσο και στον host, επιστρέφοντας τον μέσο όρο 5 μετρικών CPU ανά host, ανά ώρα, για μια μέρα.
- double-groupby-all: Aggregate τόσο στον χρόνο όσο και στον host, επιστρέφοντας τον μέσο όρο όλων των μετρικών CPU ανά host, ανά ώρα, για μια μέρα.
- high-cpu-all: Όλες οι μετρήσεις όπου μια μετρική είναι πάνω από ένα όριο για όλους τους hosts.
- high-cpu-1: Όλες οι μετρήσεις όπου μια μετρική είναι πάνω από το όριο για 1 συγκεκριμένο host.
- lastpoint: Η τελευταία μέτρηση για κάθε host.
- group-by-order-by-limit: Οι πέντε τελευταίες aggregate μετρήσεις (κατά τη διάρκεια του χρόνου) πριν από ένα τυχαία επιλεγμένο τελικό σημείο.

Β. Πειραματικά αποτελέσματα

Παρατίθενται στη συνέχεια τα αποτελέσματα των μετρήσεων μας, υπό μορφή πίνακα, για κάθε μία εκ των υπό εξέταση μετρικών για τα αρχεία small, medium, big διαδοχικά, διατεταγμένα με τη σειρά που αναλύθηκαν στην προηγούμενη παράγραφο.

Wall Clock Time (s) per worker

Query Type	Database	1 Worker	2 Workers	4 Workers
small_sgb_1_1_1	Influx	1.53 (74%)	0.93 (77%)	0.59 (84%)
	Timescale	2.08	1.21	0.7
small_sgb_1_1_12	Influx	8.04 (95%)	4.79 (52%)	3.39 (69%)
	Timescale	8.5	9.17	4.91
small_sgb_1_8_1	Influx	3.12 (98%)	1.96 (120%)	1.56 (168%)
	Timescale	3.18	1.64	0.93
small_sgb_5_1_1	Influx	3.63 (150%)	2.18 (158%)	1.75 (227%)
	Timescale	2.42	1.38	0.77
small_sgb_5_1_12	Influx	27.83 (343%)	16.53 (151%)	12.2 (214%)
	Timescale	8.12	10.97	5.7
small_sgb_5_8_1	Influx	10.14 (263%)	6.03 (308%)	5.32 (493%)
	Timescale	3.85	2.01	1.08
small_cpu_max_all_1	Influx	5.28 (76%)	3.21 (40%)	2.05 (49%)
	Timescale	6.98	8.01	4.19
small_cpu_max_all_8	Influx	14.19 (63%)	10.15 (88%)	10.06 (172%)
	Timescale	22.41	11.51	5.85
small_dgb_1	Influx	7.92 (33%)	4.18 (35%)	2.68 (46%)
	Timescale	23.74	11.79	5.84
small_dgb_5	Influx	31.61 (103%)	17.11 (109%)	9.58 (120%)
	Timescale	30.76	15.68	7.97
small_dgb_all	Influx	61.27 (156%)	33.2 (169%)	18.73 (184%)
	Timescale	39.2	19.68	10.19
small_high_cpu_all	Influx	87.84 (628%)	47.43 (1083%)	29.31 (1106%)
	Timescale	13.87	4.38	2.65
small_high_cpu_1	Influx	9.88 (192%)	5.44 (286%)	3.46 (296%)
	Timescale	5.15	1.9	1.17
small_lastpoint	Influx	3.94 (1231%)	2.24 (679%)	1.66 (1038%)
	Timescale	0.32	0.33	0.16
small_groupby_orderby_limit	Influx	32.08 (3770%)	18.3 (3660%)	16.99 (5481%)
	Timescale	0.85	0.5	0.31

Queries per second per worker

Query Type	Database	1 Worker	2 Workers	4 Workers
small_sgb_1_1_1	Influx	656.42 (137%)	1089.98 (132%)	1697.78 (118%)
	Timescale	480.62	825.89	1435.48
small_sgb_1_1_12	Influx	124.63 (106%)	209.47 (192%)	296.55 (145%)
	Timescale	118.0	109.19	204.07
small_sgb_1_8_1	Influx	320.87 (102%)	513.1 (84%)	646.94 (68%)
	Timescale	315.14	612.18	1082.64
small_sgb_5_1_1	Influx	276.76 (67%)	461.8 (64%)	575.68 (44%)
	Timescale	413.47	724.21	1312.05
small_sgb_5_1_12	Influx	35.96 (29%)	60.57 (66%)	82.12 (47%)
	Timescale	123.18	91.26	175.84
small_sgb_5_8_1	Influx	98.73 (38%)	166.27 (33%)	188.4 (20%)
	Timescale	260.02	498.84	933.87
small_cpu_max_all_1	Influx	189.53 (132%)	312.44 (250%)	491.57 (206%)
	Timescale	143.25	124.88	239.18
small_cpu_max_all_8	Influx	70.59 (158%)	98.67 (114%)	99.59 (58%)
	Timescale	44.65	86.91	171.09
small_dgb_1	Influx	126.38 (300%)	239.61 (282%)	374.27 (218%)
	Timescale	42.15	84.9	171.52
small_dgb_5	Influx	31.65 (97%)	58.5 (92%)	100.68 (83%)
	Timescale	32.52	63.82	125.66
small_dgb_all	Influx	16.33 (64%)	30.14 (59%)	53.45 (54%)
	Timescale	25.52	50.84	98.33
small_high_cpu_all	Influx	11.49 (16%)	21.09 (9%)	34.15 (9%)
	Timescale	72.12	229.7	378.61
small_high_cpu_1	Influx	101.4 (52%)	184.13 (35%)	298.13 (34%)
	Timescale	194.12	525.27	868.23
small_lastpoint	Influx	254.52 (8%)	448.35 (14%)	605.36 (9%)
	Timescale	3180.18	3143.7	6584.26
small_groupby_orderby_limit	Influx	31.21 (3%)	54.71 (3%)	58.92 (2%)
	Timescale	1177.26	2019.93	3189.8

Wall Clock Time (s) per worker

Query Type	Database	1 Worker	2 Workers	4 Workers
medium_sgb_1_1_1	Influx	1.58 (63%)	0.91 (80%)	0.67 (105%)
	Timescale	2.5	1.14	0.64
medium_sgb_1_1_12	Influx	8.24 (104%)	4.55 (120%)	3.2 (153%)
	Timescale	7.93	3.79	2.09
medium_sgb_1_8_1	Influx	3.23 (101%)	1.93 (114%)	1.5 (155%)
	Timescale	3.19	1.69	0.97
medium_sgb_5_1_1	Influx	3.72 (151%)	2.26 (177%)	1.55 (204%)
	Timescale	2.47	1.28	0.76
medium_sgb_5_1_12	Influx	29.3 (318%)	16.86 (383%)	11.38 (448%)
	Timescale	9.2	4.4	2.54
medium_sgb_5_8_1	Influx	10.5 (261%)	6.27 (317%)	5.5 (474%)
	Timescale	4.02	1.98	1.16
medium_cpu_max_all_1	Influx	5.44 (68%)	3.34 (88%)	2.15 (105%)
	Timescale	7.98	3.81	2.05
medium_cpu_max_all_8	Influx	14.76 (66%)	10.86 (96%)	10.37 (179%)
	Timescale	22.36	11.37	5.78
medium_dgb_1	Influx	8.26 (34%)	4.48 (37%)	2.7 (43%)
	Timescale	24.19	12.03	6.26
medium_dgb_5	Influx	32.2 (102%)	17.26 (110%)	9.72 (120%)
	Timescale	31.45	15.63	8.13
medium_dgb_all	Influx	62.98 (157%)	33.1 (167%)	18.89 (183%)
	Timescale	40.11	19.83	10.32
medium_high_cpu_all	Influx	108.46 (708%)	58.24 (775%)	36.56 (633%)
	Timescale	15.31	7.51	5.78
medium_high_cpu_1	Influx	12.05 (223%)	6.72 (235%)	4.34 (270%)
	Timescale	5.41	2.86	1.61
medium_lastpoint	Influx	9.54 (2271%)	5.71 (2284%)	3.85 (1674%)
	Timescale	0.42	0.25	0.23
medium_groupby_orderby_limit	Influx	115.42 (6830%)	96.88 (11304%)	94.48 (16290%)
	Timescale	1.69	0.85	0.58

Queries per second per worker

Query Type	Database	1 Worker	2 Workers	4 Workers
medium_sgb_1_1_1	Influx	636.13 (159%)	1102.31 (126%)	1515.16 (96%)
	Timescale	400.4	878.23	1577.91
medium_sgb_1_1_12	Influx	121.5 (96%)	220.57 (84%)	313.71 (66%)
	Timescale	126.26	263.94	478.38
medium_sgb_1_8_1	Influx	311.05 (99%)	520.95 (88%)	668.86 (64%)
	Timescale	313.72	593.76	1040.61
medium_sgb_5_1_1	Influx	270.36 (67%)	444.36 (57%)	648.64 (49%)
	Timescale	405.45	784.42	1329.24
medium_sgb_5_1_12	Influx	34.15 (31%)	59.39 (26%)	88.05 (22%)
	Timescale	108.76	227.4	395.63
medium_sgb_5_8_1	Influx	95.4 (38%)	159.82 (32%)	182.77 (21%)
	Timescale	248.55	506.61	868.56
medium_cpu_max_all_1	Influx	183.95 (147%)	300.85 (115%)	477.37 (97%)
	Timescale	125.31	262.73	490.52
medium_cpu_max_all_8	Influx	67.81 (152%)	92.18 (105%)	96.61 (56%)
	Timescale	44.75	88.05	173.26
medium_dgb_1	Influx	121.36 (293%)	223.62 (269%)	372.0 (233%)
	Timescale	41.36	83.19	159.9
medium_dgb_5	Influx	31.07 (98%)	58.0 (91%)	103.16 (84%)
	Timescale	31.81	64.03	123.27
medium_dgb_all	Influx	15.88 (64%)	30.24 (60%)	53.0 (55%)
	Timescale	24.94	50.47	97.13
medium_high_cpu_all	Influx	9.22 (14%)	17.18 (13%)	27.38 (16%)
	Timescale	65.35	133.31	173.37
medium_high_cpu_1	Influx	83.1 (45%)	149.17 (43%)	231.67 (37%)
	Timescale	185.02	350.47	622.21
medium_lastpoint	Influx	104.93 (4%)	175.51 (4%)	260.57 (6%)
	Timescale	2382.18	3989.39	4470.03
medium_groupby_orderby_limit	Influx	8.67 (1%)	10.41 (1%)	10.59 (1%)
	Timescale	590.78	1182.12	1752.97

Wall Clock Time (s) per worker

Query Type	Database	1 Worker	2 Workers	4 Workers
big_sgb_1_1_1	Influx	1.63 (20%)	0.96 (76%)	0.69 (92%)
	Timescale	8.31	1.26	0.75
big_sgb_1_1_12	Influx	8.46 (47%)	4.8 (126%)	3.18 (147%)
	Timescale	18.15	3.8	2.17
big_sgb_1_8_1	Influx	3.29 (86%)	2.01 (116%)	1.53 (158%)
	Timescale	3.83	1.74	0.97
big_sgb_5_1_1	Influx	3.77 (140%)	2.16 (155%)	1.66 (205%)
	Timescale	2.69	1.39	0.81
big_sgb_5_1_12	Influx	29.44 (327%)	16.82 (353%)	11.58 (449%)
	Timescale	9.0	4.77	2.58
big_sgb_5_8_1	Influx	10.48 (253%)	6.17 (298%)	5.42 (455%)
	Timescale	4.14	2.07	1.19
big_cpu_max_all_1	Influx	5.33 (48%)	3.11 (77%)	2.04 (94%)
	Timescale	11.12	4.03	2.18
big_cpu_max_all_8	Influx	14.87 (65%)	10.59 (93%)	10.58 (177%)
	Timescale	22.8	11.43	5.97
big_dgb_1	Influx	8.84 (33%)	4.36 (36%)	2.58 (40%)
	Timescale	24.61	12.27	6.38
big_dgb_5	Influx	32.7 (102%)	17.42 (110%)	9.87 (120%)
	Timescale	31.96	15.88	8.22
big_dgb_all	Influx	63.83 (158%)	33.43 (166%)	19.3 (186%)
	Timescale	40.5	20.158	10.38
big_high_cpu_all	Influx	111.4 (733%)	59.98 (731%)	37.4 (637%)
	Timescale	15.2	8.2	5.87
big_high_cpu_1	Influx	12.03 (122%)	6.58 (215%)	4.05 (228%)
	Timescale	9.87	3.06	1.78
big_lastpoint	Influx	29.57 (2987%)	17.07 (2710%)	12.03 (2673%)
	Timescale	0.99	0.63	0.45
big_groupby_orderby_limit	Influx	510.22 (9063%)	426.64 (13852%)	423.36 (25351%)
	Timescale	5.63	3.08	1.67

Queries per second per worker

Query Type	Database	1 Worker	2 Workers	4 Workers
big_sgb_1_1_1	Influx	615.98 (511%)	1053.58 (132%)	1456.37 (109%)
	Timescale	120.66	799.43	1332.29
big_sgb_1_1_12	Influx	118.47 (215%)	209.07 (79%)	315.61 (68%)
	Timescale	55.17	263.55	461.23
big_sgb_1_8_1	Influx	304.41 (116%)	504.58 (88%)	659.46 (64%)
	Timescale	261.54	575.12	1036.12
big_sgb_5_1_1	Influx	266.08 (72%)	465.63 (64%)	611.51 (49%)
	Timescale	371.46	723.01	1246.23
big_sgb_5_1_12	Influx	34.0 (31%)	59.51 (28%)	86.57 (22%)
	Timescale	111.15	210.0	388.55
big_sgb_5_8_1	Influx	95.55 (40%)	162.57 (34%)	185.04 (22%)
	Timescale	241.64	482.63	855.0
big_cpu_max_all_1	Influx	187.95 (208%)	322.42 (130%)	491.98 (107%)
	Timescale	90.22	248.11	461.28
big_cpu_max_all_8	Influx	67.34 (153%)	94.67 (108%)	94.8 (56%)
	Timescale	43.88	87.56	168.8
big_dgb_1	Influx	124.48 (306%)	230.11 (282%)	389.3 (248%)
	Timescale	40.65	81.58	157.28
big_dgb_5	Influx	30.6 (98%)	57.45 (91%)	101.57 (83%)
	Timescale	31.3	63.01	121.92
big_dgb_all	Influx	15.67 (63%)	29.94 (60%)	51.89 (54%)
	Timescale	24.7	49.65	96.66
big_high_cpu_all	Influx	8.98 (14%)	16.68 (14%)	26.76 (16%)
	Timescale	65.82	122.17	170.99
big_high_cpu_1	Influx	83.22 (82%)	152.35 (46%)	248.06 (44%)
	Timescale	101.58	328.17	567.04
big_lastpoint	Influx	33.84 (3%)	58.63 (4%)	83.31 (4%)
	Timescale	1019.37	1615.9	2325.89
big_groupby_orderby_limit	Influx	1.96 (1%)	2.34 (1%)	2.36 (0%)
	Timescale	177.84	325.25	600.23

User Time (s) per worker

Query Type	Database	1 Worker	2 Workers	4 Workers
small_sgb_1_1_1	Influx	0.18 (120%)	0.22 (157%)	0.17 (166%)
	Timescale	0.15	0.14	0.16
small_sgb_1_1_12	Influx	0.52 (68%)	0.47 (98%)	0.45 (87%)
	Timescale	0.76	0.48	0.52
small_sgb_1_8_1	Influx	0.25 (156%)	0.2 (105%)	0.22 (147%)
	Timescale	0.16	0.19	0.15
small_sgb_5_1_1	Influx	0.3 (158%)	0.26 (137%)	0.27 (142%)
	Timescale	0.19	0.19	0.19
small_sgb_5_1_12	Influx	0.87 (126%)	0.76 (96%)	0.74 (91%)
	Timescale	0.69	0.79	0.81
small_sgb_5_8_1	Influx	0.43 (205%)	0.37 (195%)	0.36 (164%)
	Timescale	0.21	0.19	0.22
small_cpu_max_all_1	Influx	0.3 (150%)	0.32 (145%)	0.32 (152%)
	Timescale	0.2	0.22	0.21
small_cpu_max_all_8	Influx	0.48 (130%)	0.42 (168%)	0.47 (235%)
	Timescale	0.37	0.25	0.2
small_dgb_1	Influx	0.45 (102%)	0.32 (100%)	0.34 (126%)
	Timescale	0.44	0.32	0.27
small_dgb_5	Influx	0.71 (137%)	0.64 (168%)	0.62 (163%)
	Timescale	0.52	0.38	0.38
small_dgb_all	Influx	0.8 (140%)	0.87 (181%)	0.78 (170%)
	Timescale	0.57	0.48	0.46
small_high_cpu_all	Influx	2.52 (87%)	2.39 (82%)	2.41 (84%)
	Timescale	2.9	2.93	2.86
small_high_cpu_1	Influx	0.59 (137%)	0.57 (121%)	0.54 (123%)
	Timescale	0.43	0.47	0.44
small_lastpoint	Influx	0.33 (275%)	0.31 (172%)	0.31 (310%)
	Timescale	0.12	0.18	0.1
small_groupby_orderby_limit	Influx	0.76 (950%)	0.5 (556%)	0.52 (520%)
	Timescale	0.08	0.09	0.1

User Time (s) per worker

Query Type	Database	1 Worker	2 Workers	4 Workers
medium_sgb_1_1_1	Influx	0.24 (114%)	0.18 (112%)	0.19 (127%)
	Timescale	0.21	0.16	0.15
medium_sgb_1_1_12	Influx	0.52 (95%)	0.43 (100%)	0.43 (100%)
	Timescale	0.55	0.43	0.43
medium_sgb_1_8_1	Influx	0.3 (143%)	0.2 (143%)	0.29 (153%)
	Timescale	0.21	0.14	0.19
medium_sgb_5_1_1	Influx	0.34 (155%)	0.28 (140%)	0.28 (140%)
	Timescale	0.22	0.2	0.2
medium_sgb_5_1_12	Influx	0.85 (109%)	0.72 (109%)	0.76 (113%)
	Timescale	0.78	0.66	0.67
medium_sgb_5_8_1	Influx	0.49 (258%)	0.4 (190%)	0.41 (216%)
	Timescale	0.19	0.21	0.19
medium_cpu_max_all_1	Influx	0.35 (113%)	0.3 (176%)	0.25 (167%)
	Timescale	0.31	0.17	0.15
medium_cpu_max_all_8	Influx	0.46 (144%)	0.46 (184%)	0.5 (208%)
	Timescale	0.32	0.25	0.24
medium_dgb_1	Influx	0.4 (95%)	0.34 (117%)	0.34 (113%)
	Timescale	0.42	0.29	0.3
medium_dgb_5	Influx	0.68 (136%)	0.64 (168%)	0.62 (148%)
	Timescale	0.5	0.38	0.42
medium_dgb_all	Influx	0.86 (141%)	0.83 (180%)	0.8 (174%)
	Timescale	0.61	0.46	0.46
medium_high_cpu_all	Influx	3.49 (86%)	3.44 (90%)	3.38 (88%)
	Timescale	4.06	3.82	3.82
medium_high_cpu_1	Influx	0.69 (115%)	0.76 (143%)	0.68 (119%)
	Timescale	0.6	0.53	0.57
medium_lastpoint	Influx	0.44 (367%)	0.4 (388%)	0.42 (280%)
	Timescale	0.12	0.13	0.15
medium_groupby_orderby_limit	Influx	0.61 (436%)	0.58 (580%)	0.64 (457%)
	Timescale	0.14	0.1	0.14

User Time (s) per worker

Query Type	Database	1 Worker	2 Workers	4 Workers
big_sgb_1_1_1	Influx	0.2 (44%)	0.22 (183%)	0.21 (105%)
	Timescale	0.45	0.12	0.2
big_sgb_1_1_12	Influx	0.57 (75%)	0.45 (98%)	0.42 (95%)
	Timescale	0.76	0.46	0.44
big_sgb_1_8_1	Influx	0.27 (123%)	0.29 (153%)	0.25 (139%)
	Timescale	0.22	0.19	0.18
big_sgb_5_1_1	Influx	0.29 (161%)	0.21 (111%)	0.28 (156%)
	Timescale	0.18	0.19	0.18
big_sgb_5_1_12	Influx	0.88 (121%)	0.72 (96%)	0.73 (107%)
	Timescale	0.73	0.75	0.68
big_sgb_5_8_1	Influx	0.45 (214%)	0.34 (189%)	0.33 (157%)
	Timescale	0.21	0.18	0.21
big_cpu_max_all_1	Influx	0.32 (74%)	0.24 (133%)	0.27 (135%)
	Timescale	0.43	0.18	0.2
big_cpu_max_all_8	Influx	0.45 (112%)	0.34 (136%)	0.44 (157%)
	Timescale	0.4	0.25	0.28
big_dgb_1	Influx	0.38 (95%)	0.3 (107%)	0.32 (110%)
	Timescale	0.4	0.28	0.29
big_dgb_5	Influx	0.67 (131%)	0.64 (160%)	0.6 (146%)
	Timescale	0.51	0.4	0.41
big_dgb_all	Influx	0.87 (140%)	0.79 (149%)	0.77 (157%)
	Timescale	0.62	0.53	0.49
big_high_cpu_all	Influx	3.57 (86%)	3.39 (84%)	3.42 (86%)
	Timescale	4.17	4.03	3.97
big_high_cpu_1	Influx	0.68 (80%)	0.64 (108%)	0.57 (106%)
	Timescale	0.85	0.59	0.54
big_lastpoint	Influx	0.57 (335%)	0.52 (260%)	0.5 (238%)
	Timescale	0.17	0.2	0.21
big_groupby_orderby_limit	Influx	0.78 (557%)	0.74 (411%)	0.79 (359%)
	Timescale	0.14	0.18	0.22

System Time (s) per worker

Query Type	Database	1 Worker	2 Workers	4 Workers
small_sgb_1_1_1	Influx	0.09 (60%)	0.06 (75%)	0.08 (100%)
	Timescale	0.15	0.08	0.08
small_sgb_1_1_12	Influx	0.12 (75%)	0.14 (78%)	0.1 (77%)
	Timescale	0.16	0.18	0.13
small_sgb_1_8_1	Influx	0.1 (71%)	0.12 (200%)	0.09 (100%)
	Timescale	0.14	0.06	0.09
small_sgb_5_1_1	Influx	0.15 (115%)	0.11 (138%)	0.1 (143%)
	Timescale	0.13	0.08	0.07
small_sgb_5_1_12	Influx	0.19 (76%)	0.2 (91%)	0.15 (94%)
	Timescale	0.25	0.22	0.16
small_sgb_5_8_1	Influx	0.13 (93%)	0.12 (150%)	0.12 (200%)
	Timescale	0.14	0.08	0.06
small_cpu_max_all_1	Influx	0.14 (82%)	0.1 (71%)	0.07 (54%)
	Timescale	0.17	0.14	0.13
small_cpu_max_all_8	Influx	0.09 (47%)	0.08 (57%)	0.08 (50%)
	Timescale	0.19	0.14	0.16
small_dgb_1	Influx	0.12 (57%)	0.12 (75%)	0.1 (62%)
	Timescale	0.21	0.16	0.16
small_dgb_5	Influx	0.16 (114%)	0.16 (94%)	0.1 (83%)
	Timescale	0.14	0.17	0.12
small_dgb_all	Influx	0.2 (111%)	0.17 (100%)	0.16 (107%)
	Timescale	0.18	0.17	0.15
small_high_cpu_all	Influx	0.3 (39%)	0.36 (106%)	0.25 (96%)
	Timescale	0.76	0.34	0.26
small_high_cpu_1	Influx	0.17 (77%)	0.16 (133%)	0.1 (71%)
	Timescale	0.22	0.12	0.14
small_lastpoint	Influx	0.1 (250%)	0.08 (200%)	0.06 (200%)
	Timescale	0.04	0.04	0.03
small_groupby_orderby_limit	Influx	0.38 (317%)	0.15 (214%)	0.14 (233%)
	Timescale	0.12	0.07	0.06

System Time (s) per worker

Query Type	Database	1 Worker	2 Workers	4 Workers
medium_sgb_1_1_1	Influx	0.08 (32%)	0.08 (100%)	0.08 (114%)
	Timescale	0.25	0.08	0.07
medium_sgb_1_1_12	Influx	0.11 (46%)	0.1 (71%)	0.12 (75%)
	Timescale	0.24	0.14	0.16
medium_sgb_1_8_1	Influx	0.07 (70%)	0.1 (91%)	0.07 (100%)
	Timescale	0.1	0.11	0.07
medium_sgb_5_1_1	Influx	0.1 (91%)	0.1 (167%)	0.08 (100%)
	Timescale	0.11	0.06	0.08
medium_sgb_5_1_12	Influx	0.16 (57%)	0.22 (100%)	0.13 (72%)
	Timescale	0.28	0.22	0.18
medium_sgb_5_8_1	Influx	0.1 (59%)	0.08 (114%)	0.06 (60%)
	Timescale	0.17	0.07	0.1
medium_cpu_max_all_1	Influx	0.11 (85%)	0.1 (91%)	0.1 (83%)
	Timescale	0.13	0.11	0.12
medium_cpu_max_all_8	Influx	0.12 (52%)	0.12 (75%)	0.08 (57%)
	Timescale	0.23	0.16	0.14
medium_dgb_1	Influx	0.17 (68%)	0.14 (88%)	0.12 (71%)
	Timescale	0.25	0.16	0.17
medium_dgb_5	Influx	0.14 (78%)	0.12 (100%)	0.09 (75%)
	Timescale	0.18	0.12	0.12
medium_dgb_all	Influx	0.15 (83%)	0.13 (87%)	0.1 (53%)
	Timescale	0.18	0.15	0.19
medium_high_cpu_all	Influx	0.36 (34%)	0.5 (62%)	0.46 (94%)
	Timescale	1.06	0.8	0.49
medium_high_cpu_1	Influx	0.19 (86%)	0.11 (61%)	0.13 (93%)
	Timescale	0.22	0.18	0.14
medium_lastpoint	Influx	0.12 (300%)	0.1 (333%)	0.08 (200%)
	Timescale	0.04	0.03	0.04
medium_groupby_orderby_limit	Influx	0.11 (110%)	0.12 (133%)	0.13 (144%)
	Timescale	0.1	0.09	0.09

System Time (s) per worker

Query Type	Database	1 Worker	2 Workers	4 Workers
big_sgb_1_1_1	Influx	0.1 (83%)	0.08 (80%)	0.06 (120%)
	Timescale	0.12	0.1	0.05
big_sgb_1_1_12	Influx	0.15 (65%)	0.13 (81%)	0.1 (83%)
	Timescale	0.23	0.16	0.12
big_sgb_1_8_1	Influx	0.08 (50%)	0.08 (133%)	0.08 (89%)
	Timescale	0.16	0.06	0.09
big_sgb_5_1_1	Influx	0.15 (136%)	0.13 (162%)	0.07 (88%)
	Timescale	0.11	0.08	0.08
big_sgb_5_1_12	Influx	0.14 (56%)	0.18 (129%)	0.14 (78%)
	Timescale	0.25	0.14	0.18
big_sgb_5_8_1	Influx	0.12 (100%)	0.1 (100%)	0.12 (150%)
	Timescale	0.12	0.1	0.08
big_cpu_max_all_1	Influx	0.1 (83%)	0.1 (91%)	0.06 (75%)
	Timescale	0.12	0.11	0.08
big_cpu_max_all_8	Influx	0.11 (92%)	0.09 (75%)	0.12 (150%)
	Timescale	0.12	0.12	0.08
big_dgb_1	Influx	0.13 (81%)	0.11 (61%)	0.11 (79%)
	Timescale	0.16	0.18	0.14
big_dgb_5	Influx	0.16 (89%)	0.12 (80%)	0.11 (110%)
	Timescale	0.18	0.15	0.1
big_dgb_all	Influx	0.2 (91%)	0.18 (120%)	0.15 (125%)
	Timescale	0.22	0.15	0.12
big_high_cpu_all	Influx	0.56 (54%)	0.51 (65%)	0.39 (98%)
	Timescale	1.03	0.78	0.4
big_high_cpu_1	Influx	0.16 (70%)	0.12 (92%)	0.13 (81%)
	Timescale	0.23	0.13	0.16
big_lastpoint	Influx	0.12 (120%)	0.12 (200%)	0.11 (550%)
	Timescale	0.1	0.06	0.02
big_groupby_orderby_limit	Influx	0.13 (76%)	0.11 (92%)	0.12 (200%)
	Timescale	0.17	0.12	0.06

CPU Usage (%) per worker

Query Type	Database	1 Worker	2 Workers	4 Workers
small_sgb_1_1_1	Influx	18 (129%)	30 (158%)	42 (124%)
	Timescale	14	19	34
small_sgb_1_1_12	Influx	7 (70%)	12 (171%)	16 (123%)
	Timescale	10	7	13
small_sgb_1_8_1	Influx	11 (122%)	10 (67%)	20 (77%)
	Timescale	9	15	26
small_sgb_5_1_1	Influx	12 (92%)	17 (89%)	21 (64%)
	Timescale	13	19	33
small_sgb_5_1_12	Influx	3 (27%)	5 (56%)	7 (41%)
	Timescale	11	9	17
small_sgb_5_8_1	Influx	5 (56%)	8 (62%)	9 (35%)
	Timescale	9	13	26
small_cpu_max_all_1	Influx	8 (160%)	13 (325%)	19 (238%)
	Timescale	5	4	8
small_cpu_max_all_8	Influx	4 (200%)	5 (167%)	5 (83%)
	Timescale	2	3	6
small_dgb_1	Influx	7 (350%)	10 (250%)	16 (229%)
	Timescale	2	4	7
small_dgb_5	Influx	2 (100%)	4 (133%)	7 (117%)
	Timescale	2	3	6
small_dgb_all	Influx	1 (100%)	3 (100%)	5 (83%)
	Timescale	1	3	6
small_high_cpu_all	Influx	3 (12%)	5 (7%)	9 (8%)
	Timescale	26	74	117
small_high_cpu_1	Influx	7 (58%)	13 (42%)	18 (36%)
	Timescale	12	31	50
small_lastpoint	Influx	11 (23%)	17 (25%)	22 (26%)
	Timescale	48	67	85
small_groupby_orderby_limit	Influx	3 (12%)	3 (9%)	3 (6%)
	Timescale	24	33	51

CPU Usage (%) per worker

Query Type	Database	1 Worker	2 Workers	4 Workers
medium_sgb_1_1_1	Influx	20 (111%)	28 (140%)	42 (124%)
	Timescale	18	20	34
medium_sgb_1_1_12	Influx	7 (70%)	11 (73%)	17 (61%)
	Timescale	10	15	28
medium_sgb_1_8_1	Influx	11 (122%)	16 (107%)	24 (89%)
	Timescale	9	15	27
medium_sgb_5_1_1	Influx	11 (85%)	17 (85%)	23 (62%)
	Timescale	13	20	37
medium_sgb_5_1_12	Influx	3 (27%)	5 (25%)	7 (21%)
	Timescale	11	20	33
medium_sgb_5_8_1	Influx	5 (56%)	7 (50%)	8 (31%)
	Timescale	9	14	26
medium_cpu_max_all_1	Influx	8 (160%)	12 (171%)	16 (123%)
	Timescale	5	7	13
medium_cpu_max_all_8	Influx	3 (150%)	5 (167%)	5 (83%)
	Timescale	2	3	6
medium_dgb_1	Influx	6 (300%)	11 (367%)	17 (243%)
	Timescale	2	3	7
medium_dgb_5	Influx	2 (100%)	4 (133%)	7 (117%)
	Timescale	2	3	6
medium_dgb_all	Influx	1 (100%)	2 (67%)	4 (67%)
	Timescale	1	3	6
medium_high_cpu_all	Influx	3 (9%)	6 (10%)	10 (14%)
	Timescale	33	61	74
medium_high_cpu_1	Influx	7 (47%)	13 (52%)	18 (42%)
	Timescale	15	25	43
medium_lastpoint	Influx	5 (13%)	8 (13%)	13 (16%)
	Timescale	39	62	80
medium_groupby_orderby_limit	Influx	0 (0%)	0 (0%)	0 (0%)
	Timescale	15	23	40

CPU Usage (%) per worker

Query Type	Database	1 Worker	2 Workers	4 Workers
big_sgb_1_1_1	Influx	19 (317%)	32 (178%)	40 (125%)
	Timescale	6	18	32
big_sgb_1_1_12	Influx	8 (160%)	12 (75%)	16 (64%)
	Timescale	5	16	25
big_sgb_1_8_1	Influx	10 (100%)	18 (120%)	21 (78%)
	Timescale	10	15	27
big_sgb_5_1_1	Influx	11 (110%)	16 (80%)	21 (66%)
	Timescale	10	20	32
big_sgb_5_1_12	Influx	3 (30%)	5 (28%)	7 (21%)
	Timescale	10	18	33
big_sgb_5_8_1	Influx	5 (62%)	7 (54%)	8 (33%)
	Timescale	8	13	24
big_cpu_max_all_1	Influx	7 (140%)	11 (157%)	16 (123%)
	Timescale	5	7	13
big_cpu_max_all_8	Influx	3 (150%)	4 (133%)	5 (83%)
	Timescale	2	3	6
big_dgb_1	Influx	6 (300%)	9 (300%)	16 (267%)
	Timescale	2	3	6
big_dgb_5	Influx	2 (100%)	4 (133%)	7 (117%)
	Timescale	2	3	6
big_dgb_all	Influx	1 (50%)	2 (67%)	4 (80%)
	Timescale	2	3	5
big_high_cpu_all	Influx	3 (9%)	6 (10%)	10 (14%)
	Timescale	34	58	74
big_high_cpu_1	Influx	7 (70%)	11 (48%)	17 (44%)
	Timescale	10	23	39
big_lastpoint	Influx	2 (7%)	3 (8%)	5 (10%)
	Timescale	28	40	51
big_groupby_orderby_limit	Influx	0 (0%)	0 (0%)	0 (0%)
	Timescale	5	9	17

Total Time (s) per worker

Load Type	Database	1 Worker	2 Workers	4 Workers	8 Workers
small_load	Influx	42.68 (96%)	21.58 (97%)	16.29 (112%)	16.46 (154%)
	Timescale	44.616	22.247	14.538	10.717

Total Time (s) per worker

Load Type	Database	1 Worker	2 Workers	4 Workers	8 Workers
medium_load	Influx	226.39 (108%)	129.77 (91%)	103.12 (120%)	97.02 (105%)
	Timescale	209.403	142.115	85.915	92.803

Total Time (s) per worker

Load Type	Database	1 Worker	2 Workers	4 Workers	8 Workers
big_load	Influx	1024.19 (108%)	597.8 (80%)	458.3 (76%)	478.17 (99%)
	Timescale	944.353	743.45	602.491	485.313

Rows per second per worker

Load Type	Database	1 Worker	2 Workers	4 Workers	8 Workers
small_load	Influx	28661.0 (105%)	56671.6 (103%)	75075.51 (89%)	74325.84 (65%)
	Timescale	27413.63	54977.7	84133.42	114127.81

Rows per second per worker

Load Type	Database	1 Worker	2 Workers	4 Workers	8 Workers
medium_load	Influx	32057.62 (92%)	55928.96 (110%)	78379.81 (83%)	74883.45 (96%)
	Timescale	34658.47	51068.67	84474.21	78284.25

Rows per second per worker

Load Type	Database	1 Worker	2 Workers	4 Workers	8 Workers
big_load	Influx	31887.78 (92%)	54632.05 (124%)	71261.89 (131%)	68299.9 (101%)
	Timescale	34583.68	43929.24	54286.98	67295.11

Metrics per second per worker

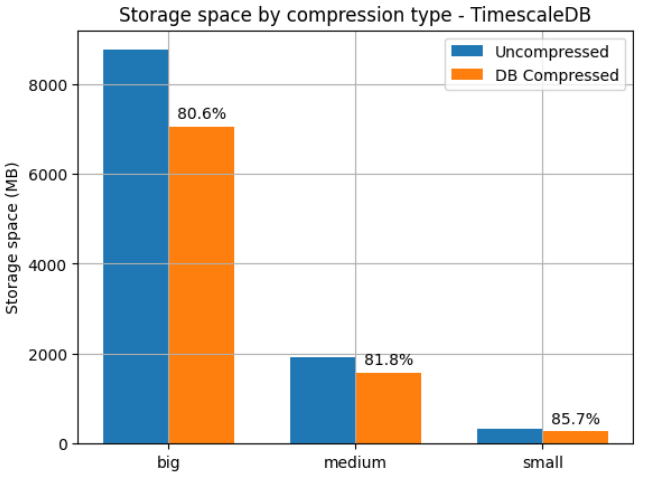
Load Type	Database	1 Worker	2 Workers	4 Workers	8 Workers
small_load	Influx	321640.06 (105%)	635981.3 (103%)	842514.07 (89%)	834101.15 (65%)
	Timescale	307641.82	616972.01	944163.93	1288767.63

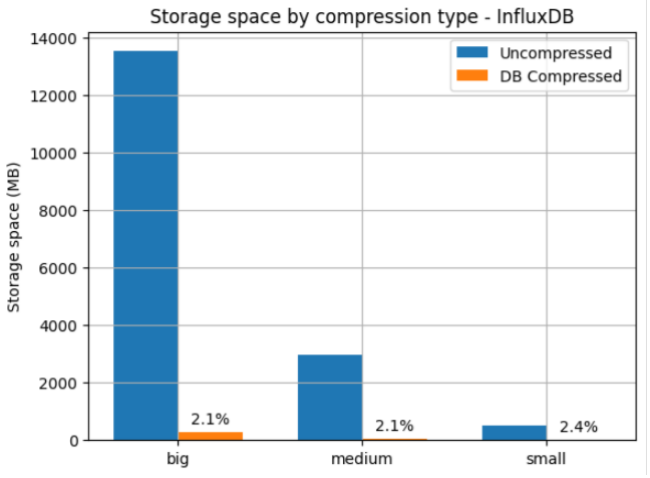
Metrics per second per worker

Load Type	Database	1 Worker	2 Workers	4 Workers	8 Workers
medium_load	Influx	359757.69 (92%)	627647.19 (110%)	789808.88 (83%)	839460.89 (96%)
	Timescale	388945.08	573103.95	947988.31	877625.47

Metrics per second per worker

Load Type	Database	1 Worker	2 Workers	4 Workers	8 Workers
big_load	Influx	357851.72 (92%)	613093.03 (124%)	799716.75 (131%)	766476.66 (101%)
	Timescale	388185.71	492983.73	608322.77	755200.66





C. Παρατηρήσεις & συμπεράσματα

Τα αποτελέσματα για τα queries αντιστοιχούν σε 1000 queries ανά query type. Επίσης, επισημαίνεται και η σχετική απόδοση του TimescaleDB σε σύγκριση με το InfluxDB, δηλώνοντας ουσιαστικά πόσο επί τοις εκατό των πόρων που χρησιμοποιεί το TimescaleDB, χρησιμοποιεί το InfluxDB. Μάλιστα, ανάλογα με τις επιθυμητές τιμές των μετρικών, η ποσοστιαία αυτή σχέση σημειώνεται με το χρώμα του «νικητή»· κόκκινο για όταν υπερτερεί το TimescaleDB έναντι του InfluxDB και πράσινο για το αντίστροφο. Προς διευκόλυνσή μας, κατηγοριοποιούμε τα queries ως εξής:

- Simple rollups: single-groupby-1-1-1/ -1-1-12/ -1-8-1/ -5-1-1/ -5-1-12/ -5-8-1
- Aggregates: cpu-max-all-1/ -8
- Double rollups: double-groupby-1/ -5/ -all
- Thresholds: high-cpu-all/ -1
- Complex queries: lastpoint, groupby-orderby-limit

Σε πρώτο στάδιο, πραγματοποιούμε σύγκριση μεταξύ των μετρικών ανά κατηγορία query ανά μέγεθος αρχείου και σε δεύτερο στάδιο εξετάζουμε την επίδραση της μεταβολής του αριθμού των workers στις τιμές των υπό εξέταση κι ανάλυση μετρικών. Συνεπώς:

1. Wall clock time
   - Simple rollups: Παρατηρούμε ότι ανεξαρτήτως του μεγέθους του αρχείου υπερτερεί το TimescaleDB. Σημειώνουμε απλώς, ότι υπερτερεί περισσότερο στο medium αρχείο, έπειτα στο big και λιγότερο στο small.
   - Aggregates: Υπερτερεί το InfluxDB.
   - Double rollups: Ενώ στο dgb_1 υπερτερεί το Influx, στα άλλα δύο υπερτερεί το TimescaleDB, άρα συνολικά υπερτερεί το TimescaleDB.
   - Thresholds: Ομόφωνα υπερτερεί το TimescaleDB.
   - Complex queries: Ομόφωνα υπερτερεί το TimescaleDB.

   Εστιάζοντας στην επίδραση των workers στις τιμές της μετρικής wall clock time, παρατηρούμε ότι η αύξηση των workers έχει ως αποτέλεσμα μεγαλύτερη μείωση στον χρόνο της TimescaleDB, απ' ότι στην InfluxDB. Αξίζει ακόμα να αναφέρουμε, ότι στα complex queries, όσο μεγαλώνει το dataset, η InfluxDB εμφανίζει σημαντική αύξηση στον χρόνο που απαιτεί (έως και αρκετά λεπτά), τη στιγμή που η TimescaleDB εμφανίζει πολύ μικρότερη αύξηση (λίγα δευτερόλεπτα).

2. Queries/second
   - Simple rollups: Παρατηρούμε ότι ανεξαρτήτως του μεγέθους του αρχείου υπερτερεί το TimescaleDB. Σημειώνουμε απλώς, ότι υπερτερεί περισσότερο στο medium αρχείο, έπειτα στο big και λιγότερο στο small.
   - Aggregates: Υπερτερεί το InfluxDB.
   - Double rollups: Ενώ στο dgb_1 υπερτερεί το Influx, στα άλλα δύο υπερτερεί το TimescaleDB, άρα συνολικά υπερτερεί το TimescaleDB.
   - Thresholds: Ομόφωνα υπερτερεί το TimescaleDB.
   - Complex queries: Ομόφωνα υπερτερεί το TimescaleDB.

   Όσον αφορά την αύξηση των workers από την άλλη, παρατηρούμε πως προκαλεί μεγαλύτερη αύξηση στα queries/second στο TimescaleDB, συγκριτικά με το InfluxDB, ενώ δεν μπορεί να παραλειφθεί το γεγονός ότι στα complex queries και στις δύο βάσεις, όσο μεγαλώνει το dataset, τόσο λιγότερα είναι τα queries/second που λαμβάνουμε.

3. User time
   - Simple rollups: Με εξαίρεση το sgb-1-1-12, ανεξαρτήτως μεγέθους, υπερτερεί το TimescaleDB.
   - Aggregates: Εμφανώς καλύτερη η απόδοση του TimescaleDB.
   - Double rollups: Εμφανώς καλύτερη η απόδοση του TimescaleDB.
   - Thresholds: Σε κάθε dataset το InfluxDB υπερτερεί στο high-cpu-all, ενώ το TimescaleDB, στο high-cpu-1.
   - Complex queries: Ολικώς υπερτερεί το TimescaleDB.

   Στην μετρική user time σημειώνουμε διαφορά επίδρασης μεταξύ της περίπτωσης του διπλασιασμού και του τετραπλασιασμού των workers. Πιο συγκεκριμένα, ο διπλασιασμός των workers, επιφέρει μια μικρή μείωση στο user time, τόσο στο InfluxDB, όσο και στο TimescaleDB, ενώ ο τετραπλασιασμός των workers δεν προσφέρει καμία επιπλέον βελτίωση.

4. System time
   - Simple rollups: Υπερτερεί το InfluxDB.
   - Aggregates: Υπερτερεί το InfluxDB.
   - Double rollups: Υπερτερεί το InfluxDB.
   - Thresholds: Υπερτερεί το InfluxDB.
   - Complex queries: Υπερτερεί το TimescaleDB.

   Αντιστοίχως, όπως και στο user time, έτσι και στο system time, παρατηρούμε ότι ο διπλασιασμός των workers, προκαλεί ελαφρώς μείωση στο system time, τόσο στο InfluxDB, όσο και στο TimescaleDB, ενόσω, ο τετραπλασιασμός των workers δεν επιφέρει καμία πρόσθετη βελτίωση.

5. CPU Usage
   - Simple rollups: Με εξαίρεση το sgb-1-1-1, το InfluxDB πλεονεκτεί σε γενικές γραμμές.
   - Aggregates: Ανεξαρτήτως μεγέθους αρχείου, το TimescaleDB υπερτερεί.
   - Double rollups: Στο dgb-1/-5 ανεξαρτήτως μεγέθους/workers το TimescaleDB είναι είτε καλύτερο είτε ίσο με το InfluxDB. Στο dgb-all ισχύει το αντίθετο.
   - Thresholds: Ξεκάθαρη υπεροχή του InfluxDB.

- Complex queries: Ξεκάθαρη υπεροχή του InfluxDB.

Στην παρούσα μετρική χρησιμοποίησης της CPU, απόρροια των πειραματικών αποτελεσμάτων είναι ότι αυξάνοντας τους workers, αυξάνεται σημαντικά το CPU Usage και στις δύο βάσεις. Αυτό φυσικά είναι λογικό, καθώς κάθε worker απαιτεί πόρους ώστε να "τρέχει" παράλληλα με τις υπόλοιπες διεργασίες, οι οποίοι πόροι μεταφράζονται κατ' επέκταση σε χρήση της CPU. Συμπερασματικά, "πολλοί workers", μεταφράζονται σε "πολλές διεργασίες που χρησιμοποιούν ταυτόχρονα την CPU".

6. Total time, Rows/second & Metrics/second

Είναι λογικό τα αποτελέσματα των τριών μετρικών για την φόρτωση δεδομένων να ταυτίζονται, καθώς όσο μικρότερος είναι ο συνολικός χρόνος, τόσες περισσότερες γραμμές και μετρικές ανά δευτερόλεπτο θα εισάγονται στις βάσεις. Σε αντίθεση με όλες τις παραπάνω μετρικές για τις οποίες έγινε σύγκριση, όπου κατά γενική ομολογία, δεν παρατηρήθηκαν συμπεριφορικές διαφοροποιήσεις ανά το μέγεθος του αρχείου, στην προκειμένη, παρατηρούνται έντονες μεταβολές που υποδηλώνει ότι στο φόρτωμα των αρχείων στην TimescaleBD και InfluxDB, καταναλώνονται σημαντικά διαφορετικοί πόροι.

- Για το small αρχείο, οι δύο βάσεις σε γενικές γραμμές φαίνεται να έχουν αντίστοιχες αποδόσεις. Ειδικότερα το InfluxDB πλεονεκτεί για τους 1 & 2 workers, ενώ το TimescaleDB υπερτερεί στους 4 & 8 workers.
- Για το medium αρχείο, σημειώνουμε ότι δεν παρατηρείται κάποιο μοτίβο αντίστοιχο του παραπάνω ανά τους workers, ενώ υπερτερεί το TimescaleDB.
- Για το big αρχείο, κατ΄ αύξηση των workers, υπερτερεί σημαντικά το InfluxDB.

Επί των μετρικών φόρτωσης δεδομένων, παρατηρούμε ότι και στα 3 datasets, η αύξηση των workers από 4 σε 8, μπορεί να προκαλέσει επιδείνωση στις τιμές των μετρικών μας. Ο λόγος που συμβαίνει αυτό, είναι γιατί το σύστημά μας έχει 4 πυρήνες, οπότε η προσθήκη παραπάνω των τεσσάρων workers μπορεί να προκαλέσει context switching overhead. Το φαινόμενο αυτό συμβαίνει γιατί οι workers απαιτούν παραπάνω πόρους από όσους το σύστημα έχει διαθέσιμους, οπότε "αναγκάζονται" να αποθηκεύσουν το state τους και να περιμένουν να ελευθερωθούν οι πόροι αυτοί, γεγονός που οδηγεί σε αξιόλογες καθυστερήσεις.

7. Data compression

Ολοκληρώνοντας, οι γραφικές παραστάσεις που παρουσιάστηκαν στο τέλος της προηγούμενης ενότητας, εκφράζουν το ποσοστό της συμπίεσης των δεδομένων, συγκρίνοντας για την ακρίβεια το μέγεθος του αρχικού dataset (unzipped) με αυτό που προέκυψε εισάγοντας το στην εκάστοτε βάση. Σημειώνουμε, λοιπόν, ότι εκ πρώτης όψεως, τα ποσοστά συμπίεσης στην InfluxDB είναι με τεράστια διαφορά μεγαλύτερα από τα ποσοστά συμπίεσης στην TimescaleDB. Παρατηρούμε δηλαδή, ότι για το big αρχείο το ποσοστό συμπίεσης είναι 19,4%, για το medium αρχείο είναι 18,2% και για το small αρχείο 14,3%, ενώ οι αντίστοιχες τιμές για την InfluxDB είναι 97,9% για το big και medium αρχείο και 97,6% για το small. Το αποτέλεσμα αυτό, ωστόσο, δεν ήταν αναμενόμενο, καθώς τόσο η TimescaleDB, όσο κι η InfluxDB χρησιμοποιούν τεχνικές συμπίεσης. Λόγω του column-oriented structure, η InfluxDB πετυχαίνει εν γένει μεγαλύτερα ποσοστά συμπίεσης. Όσον αφορά τις τεχνικές αυτές. η πρώτη χρησιμοποιεί τις δυνατότητες που προσφέρει η PostgreSQL, ενώ η δεύτερη χρησιμοποιεί αποδοτικούς αλγορίθμους συμπίεσης και κωδικοποίησης, όπως τον Snappy για συμπίεση strings και delta κωδικοποίηση για integers [6]. Ωστόσο, τα αποτελέσματα που προέκυψαν, δύναται να οφείλονται σε έναν πολυποίκιλο συνδυασμό παραγόντων, που αφορούν τον τύπο των δεδομένων, τις τεχνικές συμπίεσης που προαναφέρθηκαν, τις ρυθμίσεις των βάσεων δεδομένων, και πάνω απ' όλα, τα χαρακτηριστικά των χρονοσειριακών δεδομένων.

VII. Συνοψη

Προφανώς, για να αποφασίσουμε ποια βάση δεδομένων θα επιλέγουμε σε κάθε μας project, πρέπει να λάβουμε υπόψη πολυποίκιλους παράγοντες, όπως είναι παραδείγματος χάριν το μοντέλο των δεδομένων, τη γλώσσα των queries, την αξιοπιστία, την απόδοση κ.α.. Στη παρούσα εργασία καταφέραμε να δημιουργήσουμε δύο λειτουργικά συστήματα βάσεων δεδομένων, να τα συγκρίνουμε εστιάζοντας στο κομμάτι της απόδοσης (κυρίως latency metrics) και αποφανθήκαμε σε ποια περίπτωση υπερτερεί ποια βάση. Αποκομίσαμε, λοιπόν, την εμπειρία της δημιουργίας και της σύγκρισης, μέσω ανάλυσης των δεδομένων μας και κριτικής σκέψης.

Ο κώδικας που χρησιμοποιήθηκε, τα scripts για τα configurations των επιμέρους εργαλείων, καθώς και τα αναλυτικά αποτελέσματα των μετρικών βρίσκονται στον παρακάτω σύνδεσμο:

https://github.com/LavredisG/NTUA-Analysis-and-Design-of-Information-Systems

Αναφορες

[1] Timescale Documentation | Install TimescaleDB on Linux

[2] Install InfluxDB OSS | InfluxDB OSS v1 Documentation (influxdata.com)

[3] https://github.com/timescale/tsbs/issues/247, https://github.com/timescale/tsbs/pull/209

[4] https://github.com/timescale/timescaledb/blob/main/docs/MultiNodeDeprecation.md

[5] https://github.com/timescale/tsbs

[6] TimescaleDB vs. InfluxDB: Purpose-built for time-series data