ΠΑΝΕΠΙΣΤΗΜΙΟ ΠΕΙΡΑΙΩΣ

ΣΧΟΛΗ ΤΕΧΝΟΛΟΓΙΩΝ ΠΛΗΡΟΦΟΡΙΚΗΣ ΚΑΙ ΕΠΙΚΟΙΝΩΝΙΩΝ ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ

ΥΠΟΧΡΕΩΤΙΚΗ ΕΡΓΑΣΙΑ ΜΑΘΗΜΑΤΟΣ

ΣΥΣΤΗΜΑΤΑ ΔΙΑΧΕΙΡΙΣΗΣ ΒΑΣΕΩΝ ΔΕΔΟΜΕΝΩΝ

ΠΕΙΡΑΙΑΣ

ΦΕΒΡΟΥΑΡΙΟΣ 2024

Τεχνολογίες εργασίας	PostgreSQL 16.1 pgAdmin4 v8.2

Περιεχόμενα

Εισαγωγή 3

Παραδοχή 1 - Σειρά στηλών πινάκων 3

Παραδοχή 2 - Εισαγωγή δεδομένων στη βάση 3

Παραδοχή 3 - Στίγμα (lon, lat) 4

Παραδοχή 4 - Ερώτημα 1ο 5Β: αγκυροβολημένα πλοία 4

Ερώτημα 1ο 4

i) Βρείτε τον αριθμό των στιγμάτων (lon, lat) ανά ημερολογιακή ημέρα και ταξινομήστε το αποτέλεσμα σε φθίνουσα σειρά (ως προς το πλήθος των στιγμάτων). 5

ii) Βρείτε πόσα πλοία με ελληνική σημαία ανά τύπο πλοίου είναι καταγεγραμμένα στη ΒΔ. 7

iii) Βρείτε ποια πλοία ανέπτυξαν κάποια στιγμή ταχύτητα άνω των 30 κόμβων , τι τύπου ήταν το κάθε πλοίο και πόσα ήταν αυτά τα πλοία ανά τύπο. 8

iv) Ειδικά για τα επιβατηγά πλοία (τύποι «passenger …»), πόσα στίγματα καταγράφηκαν ανά ημέρα την περίοδο 14/08/2019 - 18/08/2019; 9

v) Ποια πλοία τύπου cargo ήταν ‘αγκυροβολημένα’ (ταχύτητα μηδέν) κάποια στιγμή μέσα στην περίοδο 15/08/2019 - 18/08/2019; Ποια για ολόκληρη την περίοδο 12/08/2019 - 19/08/2019; 11

Ερώτημα 2ο 17

Ερώτημα 3ο 35

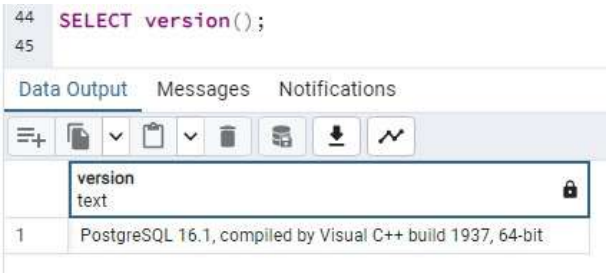
Ερώτημα 4ο – Indexing 51

Ερώτημα 5ο - Partitioning 59

Πηγές 67

Εισαγωγή

Έκδοση PostgreSQL



Έκδοση pgAdmin 4

Παραδοχή 1 - Σειρά στηλών πινάκων

Για της διευκόλυνση στην εισαγωγή των δεδομένων από τα CSV αρχεία στους πίνακες της βάσης, διατηρήθηκε η σειρά στην οποία δόθηκαν από το *DataStories* και όχι σε αυτή που παρουσιάζονται στην Εισαγωγή της εκφώνησης της εργασίας, στη σελίδα 1 του PDF.

Παραδοχή 2 - Εισαγωγή δεδομένων στη βάση

Η εισαγωγή των δεδομένων σε macOS έγινε με την χρήση της εντολής *COPY … FROM … WITH CSV HEADER* ενώ σε Windows 11 επειδή υπήρξε πρόβλημα στην εισαγωγή των δεδομένων με την συγκεκριμένη εντολή, παρόλο που δόθηκαν τα όλα απαραίτητα δικαιώματα SUPERUSER, η εισαγωγή των δεδομένων έγινε μέσω του διαχειριστικού pgAdmin όπως δείχνουν οι οδηγίες.

Παραδοχή 3 - Στίγμα (lon, lat)

Κάθε στοιχείο *lon* (longtitude) αντιστοιχεί υποχρεωτικά σε ένα στοιχείο *lat* (latitude). Επομένως, μπορούμε να ταξινομήσουμε με αύξουσα/φθίνουσα σειρά παίρνοντας την τιμή μόνο ενός εκ των δύο αφού υποχρεωτικά θα έχουν ίδιο πλήθος. Δηλαδή, στίγμα υποδηλώνει ένα *lat* που (προφανώς) αναφέρεται σε ένα *lon* και αντίθετα.

Παραδοχή 4 - Ερώτημα 1ο 5Β: αγκυροβολημένα πλοία

Ένα σκάφος θεωρείται 'αγκυροβολημένο' ακόμη και αν δεν βρεθεί εγγραφή με speed=0 για κάποια ημερομηνία. Για παράδειγμα στο dataset που μας δόθηκε, **εάν ένα σκάφος φαίνεται ακίνητο για 2 ημέρες συνεχόμενα και μετά δεν έχουμε κάποια εικόνα του στον πίνακα Positions, το θεωρούμε αγκυροβολημένο.**

Τέλος, όλοι οι χρόνοι των παρακάτω queries είναι αφού έχουμε τρέξει την εντολή VACUUM FULL; και την εντολή μια φορά ώστε να γεμίσουν οι buffers όπως αναφέρουν και οι οδηγίες της εκφώνησης.

Ερώτημα 1ο

Η κατασκευή της βάσης δεδομένων έγινε με την γνωστή σύνταξη της PostgreSQL. Το λεκτικό '*position*' είναι reserved από την PostgreSQL και για αυτό τον λόγο ονομάστηκε το table *positions*, όπως πολύ σωστά δόθηκε και από την εκφώνηση της εργασίας. Επίσης, παρόλο που παρατηρήσαμε ότι χρησιμοποιώντας τύπο *integer* ή *double precision* για τα διάφορα δεδομένα των πινάκων θα χρησιμοποιηθεί ελαφρώς λιγότερος χώρος στον δίσκο και επεξεργαστική ισχύ, από το ερώτημα (iii) που αναφέρετε σε σύγκριση πάνω στην ταχύτητα των πλοίων, αποφασίσαμε να το δηλώσουμε ως numeric διότι μας παρέχει απόλυτη ακρίβεια στις πράξεις με αυτό.

Επιτυχής φόρτωση των δεδομένων στους 3 πίνακες της βάσης:

```
26 select count(*)
27 from positions;
```

Data Output | Messages

	count bigint
1	7036651

```
26 select count(*)
27 from vessel_types;
```

Data Output | Messages

	count bigint
1	106

```
26 select count(*)
27 from vessels;
```

Data Output | Message

	count bigint
1	489

i) Βρείτε τον αριθμό των στιγμάτων (lon, lat) ανά ημερολογιακή ημέρα και ταξινομήστε το αποτέλεσμα σε φθίνουσα σειρά (ως προς το πλήθος των στιγμάτων).

```
14 SELECT t::date as time_stamp, count(lon) as longtitude, count(lat) as latitude
15 FROM positions
16 GROUP BY time_stamp
17 ORDER BY longtitude desc;
```

Data Output | Messages | Notifications

	time_stamp date	longtitude bigint	latitude bigint
1	2019-08-05	695757	695757
2	2019-08-04	543604	543604
3	2019-08-28	515392	515392
4	2019-08-06	490819	490819
5	2019-08-11	408012	408012
6	2019-08-10	406090	406090
7	2019-08-03	357393	357393
8	2019-08-29	356825	356825
9	2019-08-16	350259	350259
10	2019-08-07	341773	341773
11	2019-08-27	305732	305732
12	2019-08-17	303885	303885
13	2019-08-12	284402	284402
14	2019-08-01	253844	253844
15	2019-08-02	251133	251133
16	2019-08-15	248170	248170
17	2019-08-09	216958	216958
18	2019-08-08	207113	207113
19	2019-08-18	201904	201904
20	2019-08-19	126262	126262
21	2019-08-30	90467	90467
22	2019-08-14	72994	72994
23	2019-08-13	4427	4427
24	2019-08-26	3436	3436

Total rows: 24 of 24 Query complete 00:00:00.783

```
17 EXPLAIN ANALYZE
18 SELECT t::date as time_stamp, count(lon) as longtitude, count(lat) as latitude
19 FROM positions
20 GROUP BY time_stamp
21 ORDER BY longtitude desc;
```

Data Output | Messages | Notifications

	QUERY PLAN text
1	Sort (cost=647488.22..648624.25 rows=454411 width=20) (actual time=886.046..889.988 rows=24 loops=1)
2	Sort Key: (count(lon)) DESC
3	Sort Method: quicksort Memory: 26kB
4	-> Finalize GroupAggregate (cost=476934.60..595467.54 rows=454411 width=20) (actual time=886.009..889.975 rows=24 loops=1)
5	Group Key: ((t)::date)
6	-> Gather Merge (cost=476934.60..582971.24 rows=908822 width=20) (actual time=886.004..889.957 rows=72 loops=1)
7	Workers Planned: 2
8	Workers Launched: 2
9	-> Sort (cost=475934.58..477070.61 rows=454411 width=20) (actual time=861.638..861.640 rows=24 loops=3)
10	Sort Key: ((t)::date)
11	Sort Method: quicksort Memory: 26kB
12	Worker 0: Sort Method: quicksort Memory: 26kB
13	Worker 1: Sort Method: quicksort Memory: 26kB
14	-> Partial HashAggregate (cost=383875.11..423913.90 rows=454411 width=20) (actual time=861.434..861.618 rows=24 loops=3)
15	Group Key: (t)::date
16	Planned Partitions: 8 Batches: 1 Memory Usage: 1561kB
17	Worker 0: Batches: 1 Memory Usage: 1561kB
18	Worker 1: Batches: 1 Memory Usage: 1561kB
19	-> Parallel Seq Scan on positions (cost=0.00..165812.22 rows=2931938 width=20) (actual time=0.297..556.868 rows=2345550 loop...
20	Planning Time: 0.748 ms
21	Execution Time: 890.687 ms

Total rows: 21 of 21 | Query complete 00:00:00.949

ii) Βρείτε πόσα πλοία με ελληνική σημαία ανά τύπο πλοίου είναι καταγεγραμμένα στη ΒΔ.

```
28 SELECT type as vessel_type, count(V.id) as vessels_with_greek_flag
29 FROM vessels AS V
30 JOIN positions AS P ON V.id = P.vessel_id
31 WHERE V.flag = 'Greece'
32 GROUP BY type;
```

Data Output | Messages | Notifications

	vessel_type character varying (3)	vessels_with_greek_flag bigint
1	0	31748
2	25	33290
3	30	3477
4	31	104741
5	36	90755
6	37	340622
7	4	2953
8	40	240744
9	42	117825
10	49	183761
11	50	23628
12	51	47062

Total rows: 29 of 29 Query complete 00:00:01.071

```
9  EXPLAIN ANALYZE
10 SELECT type as vessel_type, count(V.id) as vessels_with_greek_flag
11 FROM vessels AS V
12 JOIN positions AS P ON V.id = P.vessel_id
13 WHERE V.flag = 'Greece'
14 GROUP BY type;
```

Data Output | Messages | Notifications

	QUERY PLAN text
1	Finalize GroupAggregate (cost=174577.63..174585.99 rows=33 width=10) (actual time=1218.120..1221.964 rows=29 loops=1)
2	Group Key: v.type
3	-> Gather Merge (cost=174577.63..174585.33 rows=66 width=10) (actual time=1218.114..1221.943 rows=87 loops=1)
4	Workers Planned: 2
5	Workers Launched: 2
6	-> Sort (cost=173577.60..173577.68 rows=33 width=10) (actual time=1198.690..1198.692 rows=29 loops=3)
7	Sort Key: v.type
8	Sort Method: quicksort Memory: 26kB
9	Worker 0: Sort Method: quicksort Memory: 26kB
10	Worker 1: Sort Method: quicksort Memory: 26kB
11	-> Partial HashAggregate (cost=173576.44..173576.77 rows=33 width=10) (actual time=1198.621..1198.626 rows=29 loops=3)
12	Group Key: v.type
13	Batches: 1 Memory Usage: 24kB
14	Worker 0: Batches: 1 Memory Usage: 24kB
15	Worker 1: Batches: 1 Memory Usage: 24kB
16	-> Hash Join (cost=16.16..166261.58 rows=1462971 width=67) (actual time=0.498..895.939 rows=1824966 loops=3)
17	Hash Cond: ((p.vessel_id)::text = (v.id)::text)
18	-> Parallel Seq Scan on positions p (cost=0.00..158482.38 rows=2931938 width=65) (actual time=0.199..384.841 rows=2345550 loop...
19	-> Hash (cost=13.11..13.11 rows=244 width=67) (actual time=0.282..0.283 rows=244 loops=3)
20	Buckets: 1024 Batches: 1 Memory Usage: 32kB
21	-> Seq Scan on vessels v (cost=0.00..13.11 rows=244 width=67) (actual time=0.179..0.242 rows=244 loops=3)
22	Filter: ((flag)::text = 'Greece'::text)
23	Rows Removed by Filter: 245
24	Planning Time: 0.278 ms
25	Execution Time: 1222.010 ms

iii) Βρείτε ποια πλοία ανέπτυξαν κάποια στιγμή ταχύτητα άνω των 30 κόμβων , τι τύπου ήταν το κάθε πλοίο και πόσα ήταν αυτά τα πλοία ανά τύπο.

```
6  SELECT V.id, V.type, count(V.type)
7  FROM vessels AS V
8  JOIN positions AS P ON V.id = P.vessel_id
9  WHERE P.speed > 30
10 GROUP BY V.id, V.type;
```

Data Output | Messages | Notifications

	id [PK] character varying (64)	type character varying (3)	count bigint
1	1eee5599ef5de5c07df6698ab4d4a6e6bbe4a2be25208864a8876cf6050452...	40	32971
2	26199b57ef2b787a02bf09d81b98aa7ca1a984091c6a2b6d00ee43237a253...	80	7
3	3fc28f4d2b3c7cb5f68d5b71f6fda727e4f8cd66515d98466fd3d1066e624ede	49	10859
4	4babbcb0d3d48cf988de1a317acb5a62ce72bf0a17e5510fa3d4ebdc9ff6fb7b	31	37
5	5147e5cdc9778089d8434f0a53cfb973413645f4ce59a6f4bd95decd646fa61e	[null]	0
6	53880c54896ed4abe0fc51d4173ca486a97f679a8e34b6dc3ebe4a6586bc3f...	49	5430
7	57139e85177dea04c5a366a3c24ca31a613754c1379d3e5cd844995e479dd...	42	68564
8	7475bb3647d8bb62aa76547601ddf92dc356b5de07fcddd3fcfb2c1c43e8d9...	71	2
9	8e75c1624d2d78c0f3df3598aaa14f5fcfe6252557fb7b3f8b0426d1f5d5edbc	80	4
10	9a07029a6294dcf984fba483879732a7b9cc864ef7cd70003c0f7befb8a1252d	70	15
11	9d6bce74b377738891597e999b99deaa67fd5b90466d8d40b56dbefd295d1...	40	12786
12	9ea327071a3ab8fb39601c3c7ff79397fc7de77f7fb2aa8d25daa2631400b133	37	14
13	b6ce45e37aff96757b2a87fd26dc99a7ec30482321646863e08061e01c3a6c...	40	8436
14	c37d0fca7c63e1dcd66f65485717bf7737bc352a394c3cf8bbda63c3a38b6b...	[null]	0
15	c7264ef76fbbcb3a6695a16313e0bda49aa753d1fe3a5aa625be915ad75612...	60	16
16	d8424a3a6af4a7a3aedb80308bc6a000c5a6b99511706aa04a79d32262a2b...	40	4599
17	eb538d8d3b4da934559c3bc761faf1fb6414f80e54b037d915a6172b32f7aabf	40	56212
18	f695447603e623ec3e0f9d7667932e9a81880b56757f966cfd6d647d6ce355...	40	9782

```
5  EXPLAIN ANALYZE
6  SELECT V.id, V.type, count(V.type)
7  FROM vessels AS V
8  JOIN positions AS P ON V.id = P.vessel_id
9  WHERE P.speed > 30
10 GROUP BY V.id, V.type;
```

Data Output | Messages | Notifications

	QUERY PLAN text
1	Finalize GroupAggregate (cost=167497.51..167621.40 rows=489 width=75) (actual time=606.653..609.867 rows=18 loops=1)
2	Group Key: v.id
3	-> Gather Merge (cost=167497.51..167611.62 rows=978 width=75) (actual time=606.646..609.854 rows=45 loops=1)
4	Workers Planned: 2
5	Workers Launched: 2
6	-> Sort (cost=166497.49..166498.71 rows=489 width=75) (actual time=589.716..589.718 rows=15 loops=3)
7	Sort Key: v.id
8	Sort Method: quicksort Memory: 26kB
9	Worker 0: Sort Method: quicksort Memory: 26kB
10	Worker 1: Sort Method: quicksort Memory: 26kB
11	-> Partial HashAggregate (cost=166470.75..166475.64 rows=489 width=75) (actual time=589.668..589.672 rows=15 loops=3)
12	Group Key: v.id
13	Batches: 1 Memory Usage: 49kB
14	Worker 0: Batches: 1 Memory Usage: 49kB
15	Worker 1: Batches: 1 Memory Usage: 49kB
16	-> Hash Join (cost=18.00..166051.98 rows=83754 width=67) (actual time=1.267..576.092 rows=69914 loops=3)
17	Hash Cond: ((p.vessel_id)::text = (v.id)::text)
18	-> Parallel Seq Scan on positions p (cost=0.00..165812.22 rows=83754 width=65) (actual time=0.899..558.625 rows=69914 loops=3)
19	Filter: (speed > '30'::numeric)
20	Rows Removed by Filter: 2275837
21	-> Hash (cost=11.89..11.89 rows=489 width=67) (actual time=0.351..0.351 rows=489 loops=3)
22	Buckets: 1024 Batches: 1 Memory Usage: 56kB
23	-> Seq Scan on vessels v (cost=0.00..11.89 rows=489 width=67) (actual time=0.191..0.255 rows=489 loops=3)
24	Planning Time: 0.328 ms
25	Execution Time: 609.939 ms

iv) Ειδικά για τα επιβατηγά πλοία (τύποι «passenger …»), πόσα στίγματα καταγράφηκαν ανά ημέρα την περίοδο 14/08/2019 - 18/08/2019;

```
35 SELECT P.t::date AS given_day, count(lon) AS registered_vessel_pisition
36 FROM positions AS P
37 JOIN vessels AS V on P.vessel_id = V.id
38 JOIN vessel_types AS VT ON  V.type = VT.code
39 WHERE VT.description LIKE 'Passenger,%'
40     AND P.t::date BETWEEN '2019-08-14' AND '2019-08-18'
41 GROUP BY P.t::date;
```

Data Output | Messages | Notifications

	given_day date	registered_vessel_pisition bigint
1	2019-08-14	20933
2	2019-08-15	69960
3	2019-08-16	109647
4	2019-08-17	97092
5	2019-08-18	63861

Total rows: 5 of 5 | Query complete 00:00:00.652

```
4  EXPLAIN ANALYZE
5  SELECT P.t::date AS given_day, count(lon) AS registered_vessel_pisition
6  FROM positions AS P
7  JOIN vessels AS V on P.vessel_id = V.id
8  JOIN vessel_types AS VT ON  V.type = VT.code
9  WHERE VT.description LIKE 'Passenger,%'
10     AND P.t::date BETWEEN '2019-08-14' AND '2019-08-18'
11 GROUP BY P.t::date;
```

Data Output | Messages | Notifications

	QUERY PLAN text
1	Finalize GroupAggregate (cost=188962.40..189364.64 rows=3319 width=12) (actual time=622.428..637.868 rows=5 loops=1)
2	Group Key: ((p.t)::date)
3	-> Gather Merge (cost=188962.40..189309.32 rows=2766 width=12) (actual time=617.827..637.846 rows=15 loops=1)
4	Workers Planned: 2
5	Workers Launched: 2
6	-> Partial GroupAggregate (cost=187962.37..187990.03 rows=1383 width=12) (actual time=568.448..587.703 rows=5 loops=3)
7	Group Key: ((p.t)::date)
8	-> Sort (cost=187962.37..187965.83 rows=1383 width=12) (actual time=567.420..576.871 rows=120498 loops=3)
9	Sort Key: ((p.t)::date)
10	Sort Method: external merge Disk: 2656kB
11	Worker 0: Sort Method: external merge Disk: 2720kB
12	Worker 1: Sort Method: external merge Disk: 2688kB
13	-> Hash Join (cost=16.24..187890.22 rows=1383 width=12) (actual time=64.595..547.918 rows=120498 loops=3)
14	Hash Cond: ((p.vessel_id)::text = (v.id)::text)
15	-> Parallel Seq Scan on positions p (cost=0.00..187801.76 rows=14660 width=81) (actual time=63.484..489.816 rows=392404 loop...
16	Filter: (((t)::date >= '2019-08-14'::date) AND ((t)::date <= '2019-08-18'::date))
17	Rows Removed by Filter: 1953146
18	-> Hash (cost=15.67..15.67 rows=46 width=65) (actual time=0.873..0.875 rows=64 loops=3)
19	Buckets: 1024 Batches: 1 Memory Usage: 15kB
20	-> Hash Join (cost=2.45..15.67 rows=46 width=65) (actual time=0.748..0.852 rows=64 loops=3)
21	Hash Cond: ((v.type)::text = (vt.code)::text)
22	-> Seq Scan on vessels v (cost=0.00..11.89 rows=489 width=67) (actual time=0.367..0.412 rows=489 loops=3)
23	-> Hash (cost=2.33..2.33 rows=10 width=2) (actual time=0.370..0.370 rows=10 loops=3)
24	Buckets: 1024 Batches: 1 Memory Usage: 9kB
25	-> Seq Scan on vessel_types vt (cost=0.00..2.33 rows=10 width=2) (actual time=0.349..0.354 rows=10 loops=3)
26	Filter: ((description)::text ~~ 'Passenger,%'::text)
27	Rows Removed by Filter: 96
28	Planning Time: 0.501 ms
29	Execution Time: 639.614 ms

v) Ποια πλοία τύπου cargo ήταν ‘αγκυροβολημένα’ (ταχύτητα μηδέν) κάποια στιγμή μέσα στην περίοδο 15/08/2019 - 18/08/2019; Ποια για ολόκληρη την περίοδο 12/08/2019 - 19/08/2019;

5A) Για την κατασκευή του πρώτου ερωτήματος χρησιμοποιήθηκε σύνδεση πάνω στο *vessel_id* το οποίο είναι μοναδικό στον πίνακα *Vessels*. Αρχικά το ερώτημα υλοποιήθηκε με το keyword *DISTINCT V.id* αλλά στην πορεία διαπιστώθηκε ότι μια πρώτη βελτιστοποίηση θα ήταν να μετατραπεί σε *GROUP BY* clause καθώς όπως είδαμε και στα εργαστήρια του μαθήματος είναι πιο αποδοτικό.

```
SELECT V.id as vessel_id, P.speed as stationary, V.type as cargo_vessels
FROM positions AS P
JOIN vessels AS V on P.vessel_id = V.id WHERE
V.type::integer BETWEEN 70 and 79
    AND P.speed = 0
    AND P.t BETWEEN '2019-08-15' AND '2019-08-18'
GROUP BY V.id, P.speed
```

5B) Για την κατασκευή της δεύτερης επερώτησης κατασκευάζουμε ένα βοηθητικό ερώτημα ως εξής:

```
SELECT P.t AS time_stamp, V.id, P.speed AS speed
FROM positions AS P
JOIN vessels AS V on P.vessel_id = V.id
WHERE V.type::integer BETWEEN 70 and 79
        AND P.t BETWEEN '2019-08-12' AND '2019-08-19'
    AND V.id = '05edaaf038e80b4952faf8ea6ee12f2e95066896dc0bc91fdaea37c49124685a'
```

Με αυτό τον τρόπο μπορούμε πολύ εύκολα να εξετάσουμε **<u>όλες τις εγγεγραμένες ταχύτητες</u>**, στον πίνακα positions, για ένα συγκεκριμένο πλοίο (*vessel_id*) για την ζητούμενη χρονική περίοδο. Παρατηρούμε ότι για τα διάφορα vessel_id του αποτελέσματος του προηγούμενου επερωτήματος, το συγκεκριμένο query μας επιστρέφει ταχύτητες που είναι μεγαλύτερες του 0. Για παράδειγμα για το vessel_id:

‘04b6c84a518f833b7206a114ada7660d56888de1af6cabc7bbe2c46ea6a7d733’ το οποίο είναι το πρώτο από τα 30 αποτελέσματα που επιστρέφει το **5A** επερώτημα, έχουμε:

Data Output | Messages | Notifications

	time_stamp date	id character varying (64)	speed numeric (4,1)
1	2019-08-15	04b6c84a518f833b7206a114ada7660d56888de1af6cabc7bbe2c46ea6a7d7...	0.7
2	2019-08-15	04b6c84a518f833b7206a114ada7660d56888de1af6cabc7bbe2c46ea6a7d7...	10.2
3	2019-08-15	04b6c84a518f833b7206a114ada7660d56888de1af6cabc7bbe2c46ea6a7d7...	10.2
4	2019-08-15	04b6c84a518f833b7206a114ada7660d56888de1af6cabc7bbe2c46ea6a7d7...	10.2
5	2019-08-15	04b6c84a518f833b7206a114ada7660d56888de1af6cabc7bbe2c46ea6a7d7...	10.2
6	2019-08-15	04b6c84a518f833b7206a114ada7660d56888de1af6cabc7bbe2c46ea6a7d7...	10.2
7	2019-08-15	04b6c84a518f833b7206a114ada7660d56888de1af6cabc7bbe2c46ea6a7d7...	10.2
8	2019-08-15	04b6c84a518f833b7206a114ada7660d56888de1af6cabc7bbe2c46ea6a7d7...	10.2
9	2019-08-15	04b6c84a518f833b7206a114ada7660d56888de1af6cabc7bbe2c46ea6a7d7...	10.2
10	2019-08-15	04b6c84a518f833b7206a114ada7660d56888de1af6cabc7bbe2c46ea6a7d7...	10.2

Αυτομάτως συμπεραίνουμε ότι το σκάφος φάνηκε για μικρό διάστημα στάσιμο αλλά όχι για ολόκληρο το διάστημα 12/08/2019-19/09/2019. Το ίδιο συμβαίνει σε τυχαία vessel_id που δοκιμάστηκαν από το σύνολο των 30. Συνοψίζοντας, από τα 30 αποτελέσματα του προηγούμενου επερωτήματος, πολύ λιγότερα θα ανήκουν στο αποτέλεσμα της **5Β** ερώτησης. Πράγματι, το παρακάτω query που απαντάει την συγκεκριμένη ερώτηση

```
SELECT P.vessel_id as completely_stationary_vessels_between_12_and_18_August2019
FROM positions AS P
JOIN vessels AS V on P.vessel_id = V.id
WHERE V.type::integer BETWEEN 70 and 79
    AND P.t::date BETWEEN '2019-08-12' AND '2019-08-19'
GROUP BY P.vessel_id
HAVING MAX(P.speed) = 0;
```

μας επιστρέφει 1 μοναδικό vessel_id *"9a07029a6294dcf984fba483879732a7b9cc864ef7cd70003c0f7befb8a1252d",* το οποίο αν το εισάγουμε στο παραπάνω βοηθητικό query, μας επιστρέφει 3968 διαφορετικά *timestamps* στα οποία το συγκεκριμένο σκάφος είχε ταχύτητα μηδενική κάτι το οποίο επαληθεύει περαιτέρω το αποτέλεσμα μας. Τα στίγματα του ξεκινάνε 12/08/2023 και σταματάνε 13/08/2023 ωστόσο εμείς θεωρούμε (βλ. Παραδοχή 3) ότι παρέμεινε στάσιμο καθ' όλη τη διάρκεια που μας ζητείτε.

```
28 SELECT P.t::date AS time_stamp, V.id, P.speed AS speed
29 FROM positions AS P
30 JOIN vessels AS V on P.vessel_id = V.id
31 WHERE V.type::integer BETWEEN 70 and 79
32     AND P.t BETWEEN '2019-08-12' AND '2019-08-19'
33     AND V.id = '9a07029a6294dcf984fba483879732a7b9cc864ef7cd70003c0f7befb8a1
34 ORDER BY P.t::date ASC
35
```

Data Output | Messages | Notifications

	time_stamp date	id character varying (64)	speed numeric (4,1)
1	2019-08-12	9a07029a6294dcf984fba483879732a7b9cc864ef7cd70003c0f7befb8a125...	0.0
2	2019-08-12	9a07029a6294dcf984fba483879732a7b9cc864ef7cd70003c0f7befb8a125...	0.0
3	2019-08-12	9a07029a6294dcf984fba483879732a7b9cc864ef7cd70003c0f7befb8a125...	0.0
4	2019-08-12	9a07029a6294dcf984fba483879732a7b9cc864ef7cd70003c0f7befb8a125...	0.0
5	2019-08-12	9a07029a6294dcf984fba483879732a7b9cc864ef7cd70003c0f7befb8a125...	0.0
6	2019-08-12	9a07029a6294dcf984fba483879732a7b9cc864ef7cd70003c0f7befb8a125...	0.0
7	2019-08-12	9a07029a6294dcf984fba483879732a7b9cc864ef7cd70003c0f7befb8a125...	0.0
8	2019-08-12	9a07029a6294dcf984fba483879732a7b9cc864ef7cd70003c0f7befb8a125...	0.0
9	2019-08-12	9a07029a6294dcf984fba483879732a7b9cc864ef7cd70003c0f7befb8a125...	0.0
10	2019-08-12	9a07029a6294dcf984fba483879732a7b9cc864ef7cd70003c0f7befb8a125...	0.0

Total rows: 1000 of 3968 | Query complete 00:00:00.494

Άρα έχουμε ως αποτελέσματα και τελικούς χρόνους τα παρακάτω:

5A)

```
5  SELECT V.id as vessel_id, P.speed as stationary, V.type as cargo_vessels
6  FROM positions AS P
7  JOIN vessels AS V on P.vessel_id = V.id
8  WHERE V.type::integer BETWEEN 70 and 79
9      AND P.speed = 0
10     AND P.t BETWEEN '2019-08-15' AND '2019-08-18'
11 GROUP BY V.id, P.speed
```

Data Output | Messages | Notifications

	vessel_id character varying (64)	stationary numeric (4,1)	cargo_vessels character varying (3)
1	04b6c84a518f833b7206a114ada7660d56888de1af6cabc7bbe2c46ea6a7d7...	0.0	70
2	05edaaf038e80b4952faf8ea6ee12f2e95066896dc0bc91fdaea37c49124685a	0.0	70
3	065809b408897ba8c70edf8ebd41aa8a52517815c144f952f25bc20eb79c67...	0.0	70
4	0d90e6842e1454ddcbc14e7832be23e7b4472c442e9598c6b81aad72f5828...	0.0	70
5	0f716f97f000431fee902e552d9f059fe7c126268f8ceeb65ac26205c5b659f6	0.0	70
6	404a6c642206c3340085e40ce8f8a18ad8ddafca1cdbf6e76e495a58bba0f624	0.0	71
7	40de79bc837859e58b9d405f223ab4d381a9fceea9c3eb8520f18ee99f19d6a3	0.0	72
8	40e17976df71c882d804a8bdd16af62ba28b3ae60fcb482dde89122e5059d0...	0.0	70
9	484148639e0c8229927fb9da7747a72cf0b31b56aa8a1c9d1371d9a63603d...	0.0	74
10	5b53cea1893df61389767637884d1188ca8a9da7ebce5b70254cf8e8fa1f7bdc	0.0	70
11	60c934a8d1afcc7864867ae95dd1532eab89f280fcd0029d96a283f93a09891e	0.0	71
12	66d16aac6efd45973438ed0f717eb8a1d0d4d40a3fd40dec11e3b9ec40730b...	0.0	70
13	69cb1c8b3a1d7c375ab3905cd395a2ddb30ae7ecdee1b32182644805d0a02...	0.0	71
14	6bbe3d8b3472cf97c4b7265236af5ab6298f07063e33d8baa84a333722c8ac...	0.0	73
15	7475bb3647d8bb62aa76547601ddf92dc356b5de07fcddd3fcfb2c1c43e8d9be	0.0	71
16	78557b0569d70a6dd7d599e26dab800e48b165ad79c261e1ee7243db7f06c...	0.0	70
17	7bf6ebddc7d70538694566e7891fe15cad8df821fdb5f8f0187701345e7957d8	0.0	70
18	80bc912679dfb6ff8574a0876f2ec400a43a5f4faef41d76c4345d28ca27c91a	0.0	71
19	8d56a28f51163c4b3b34f8bb2ae7f419940800e9b053f7e2ab262a8f5d22d00f	0.0	73
20	8dfb0b67f2fbc00cdc68b3ba9611fc2ba25d8314ad9370de9be43f3fd128530b	0.0	71
21	9001bc01dfc3c35a12c634d164749895bc0898664dd653fe86c51b99741f0fa2	0.0	70
22	9461996576cf9ba45b77da6b0ffd3a68baf2634fb1803b958862f72bb879873a	0.0	70
23	a22a17d47bff1c7508678f42bb840b7ef3410697fadca9903dd8097012531d4f	0.0	70
24	b3f3ee37465e4aa496cc55d533a52a2baee57834d986e5e6fd0701d0d85d6c...	0.0	71
25	bac263db165c5cef0003ad51e8d9161990f7c2e9f9fab89feb46289f58a35b0d	0.0	71
26	bf6e7816d1c33fe2d9afc3a9fca90210f521eb702655efb86e4621ab14a38e22	0.0	71
27	d030cef60611b7dfe659b8e93f4f0f3fdb26ed974229f6c6b47396acc5f07381	0.0	71
28	d6b4ca21a3677d0cc9d9f8f3bcae8138af5a54c52eefd8f7935eb09e93cd3a16	0.0	70
29	da8ce411f6eb032f235332e1ece0dc61e8022bf8e18cebc06de9879e97f1f8c2	0.0	74
30	e3d1c069fc27183c557a8665d1588024df55b4655257823c1d7c41e597f743...	0.0	70

```
5  EXPLAIN ANALYZE
6  SELECT V.id as vessel_id, P.speed as stationary, V.type as cargo_vessels
7  FROM positions AS P
8  JOIN vessels AS V on P.vessel_id = V.id
9  WHERE V.type BETWEEN '70' and '79'
10     AND P.speed = 0.0
11     AND P.t >= '2019-08-15 00:00:00' AND P.t <= '2019-08-18 00:00:00'
12 GROUP BY V.id, P.speed;
```

Data Output | Messages | Notifications

	QUERY PLAN text
1	Group (cost=182156.46..182181.49 rows=105 width=71) (actual time=518.643..522.110 rows=30 loops=1)
2	Group Key: v.id
3	-> Gather Merge (cost=182156.46..182180.96 rows=210 width=71) (actual time=518.642..522.097 rows=89 loops=1)
4	Workers Planned: 2
5	Workers Launched: 2
6	-> Sort (cost=181156.44..181156.70 rows=105 width=71) (actual time=500.146..500.148 rows=30 loops=3)
7	Sort Key: v.id
8	Sort Method: quicksort Memory: 27kB
9	Worker 0: Sort Method: quicksort Memory: 27kB
10	Worker 1: Sort Method: quicksort Memory: 27kB
11	-> Partial HashAggregate (cost=181151.86..181152.91 rows=105 width=71) (actual time=500.066..500.072 rows=30 loops=3)
12	Group Key: v.id
13	Batches: 1 Memory Usage: 24kB
14	Worker 0: Batches: 1 Memory Usage: 24kB
15	Worker 1: Batches: 1 Memory Usage: 24kB
16	-> Hash Join (cost=15.65..181039.89 rows=44791 width=71) (actual time=225.157..498.547 rows=7094 loops=3)
17	Hash Cond: ((p.vessel_id)::text = (v.id)::text)
18	-> Parallel Seq Scan on positions p (cost=0.00..160471.91 rows=208600 width=69) (actual time=195.802..482.096 rows=160205 loops=3)
19	Filter: ((t >= '2019-08-15 00:00:00'::timestamp without time zone) AND (t <= '2019-08-18 00:00:00'::timestamp without time zone) AND (speed = 0.0))
20	Rows Removed by Filter: 2165346
21	-> Hash (cost=14.34..14.34 rows=105 width=67) (actual time=0.521..0.522 rows=104 loops=3)
22	Buckets: 1024 Batches: 1 Memory Usage: 19kB
23	-> Seq Scan on vessels v (cost=0.00..14.34 rows=105 width=67) (actual time=0.338..0.499 rows=104 loops=3)
24	Filter: (((type)::text >= '70'::text) AND ((type)::text <= '79'::text))
25	Rows Removed by Filter: 385
26	Planning Time: 0.306 ms
27	Execution Time: 522.240 ms

5B)

```
40 SELECT P.vessel_id as completely_stationary_vessels_betweeen_12_and_18_August2019
41 FROM positions AS P
42 JOIN vessels AS V on P.vessel_id = V.id
43 WHERE V.type::integer BETWEEN 70 and 79
44     AND P.t::date BETWEEN '2019-08-12' AND '2019-08-19'
45 GROUP BY P.vessel_id
46 HAVING MAX(P.speed) = 0;
```

Data Output | Messages | Notifications

	completely_stationary_vessels_betweeen_12_and_18_august2019 character varying (64)
1	9a07029a6294dcf984fba483879732a7b9cc864ef7cd70003c0f7befb8a125...

Total rows: 1 of 1 | Query complete 00:00:01.776

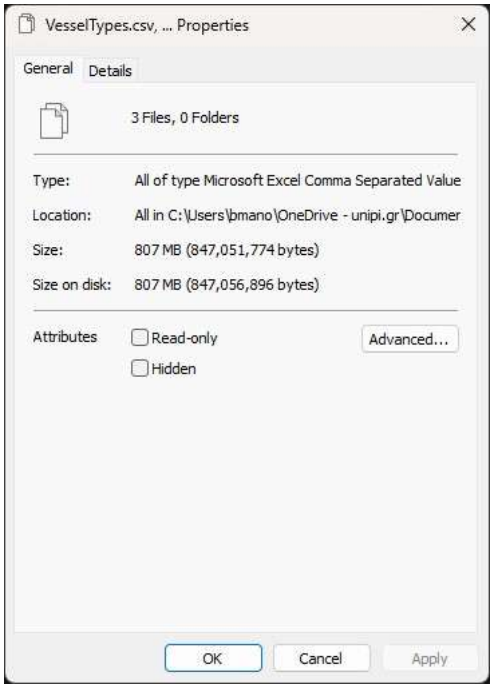
```sql
EXPLAIN ANALYZE
SELECT P.vessel_id as completely_stationary_vessels_between_12_and_18_August2019
FROM positions AS P
JOIN vessels AS V on P.vessel_id = V.id
WHERE V.type::integer BETWEEN 70 and 79
    AND P.t >= '2019-08-12 00:00:00' AND P.t <= '2019-08-19 00:00:00'
GROUP BY P.vessel_id
HAVING MAX(P.speed) = 0.0;
```

Data Output | Messages | Notifications

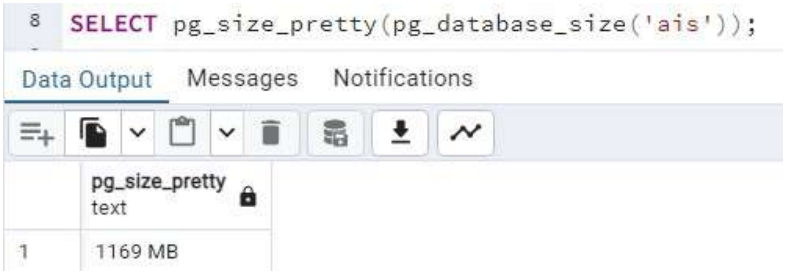
	QUERY PLAN text
1	Finalize GroupAggregate (cost=175942.68..176072.80 rows=2 width=65) (actual time=597.880..599.907 rows=1 loops=1)
2	Group Key: p.vessel_id
3	Filter: (max(p.speed) = 0.0)
4	Rows Removed by Filter: 48
5	-> Gather Merge (cost=175942.68..176063.30 rows=844 width=97) (actual time=593.188..599.860 rows=143 loops=1)
6	Workers Planned: 2
7	Workers Launched: 2
8	-> Partial GroupAggregate (cost=174942.66..174965.86 rows=422 width=97) (actual time=562.824..570.156 rows=48 loops=3)
9	Group Key: p.vessel_id
10	-> Sort (cost=174942.66..174948.99 rows=2531 width=69) (actual time=562.757..565.009 rows=29614 loops=3)
11	Sort Key: p.vessel_id
12	Sort Method: quicksort Memory: 3531kB
13	Worker 0: Sort Method: quicksort Memory: 3439kB
14	Worker 1: Sort Method: quicksort Memory: 3664kB
15	-> Hash Join (cost=19.25..174799.59 rows=2531 width=69) (actual time=34.807..530.953 rows=29614 loops=3)
16	Hash Cond: ((p.vessel_id)::text = (v.id)::text)
17	-> Parallel Seq Scan on positions p (cost=0.00..173142.07 rows=618741 width=69) (actual time=30.328..476.976 rows=48868...
18	Filter: ((t >= '2019-08-12 00:00:00'::timestamp without time zone) AND (t <= '2019-08-19 00:00:00'::timestamp without time z...
19	Rows Removed by Filter: 1856870
20	-> Hash (cost=19.23..19.23 rows=2 width=65) (actual time=0.410..0.411 rows=104 loops=3)
21	Buckets: 1024 Batches: 1 Memory Usage: 18kB
22	-> Seq Scan on vessels v (cost=0.00..19.23 rows=2 width=65) (actual time=0.316..0.388 rows=104 loops=3)
23	Filter: (((type)::integer >= 70) AND ((type)::integer <= 79))
24	Rows Removed by Filter: 385
25	Planning Time: 0.296 ms
26	Execution Time: 600.300 ms

Ερώτημα 2ο

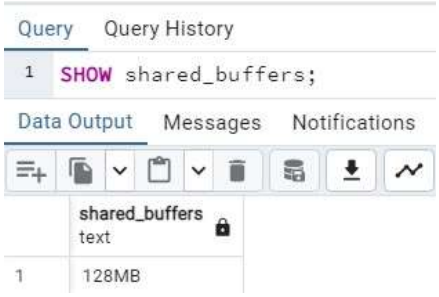
Παρατηρούμε ότι ολόκληρο το dataset που μας δίνεται είναι περίπου 800MB (uncompressed) όπως φαίνεται παρακάτω:



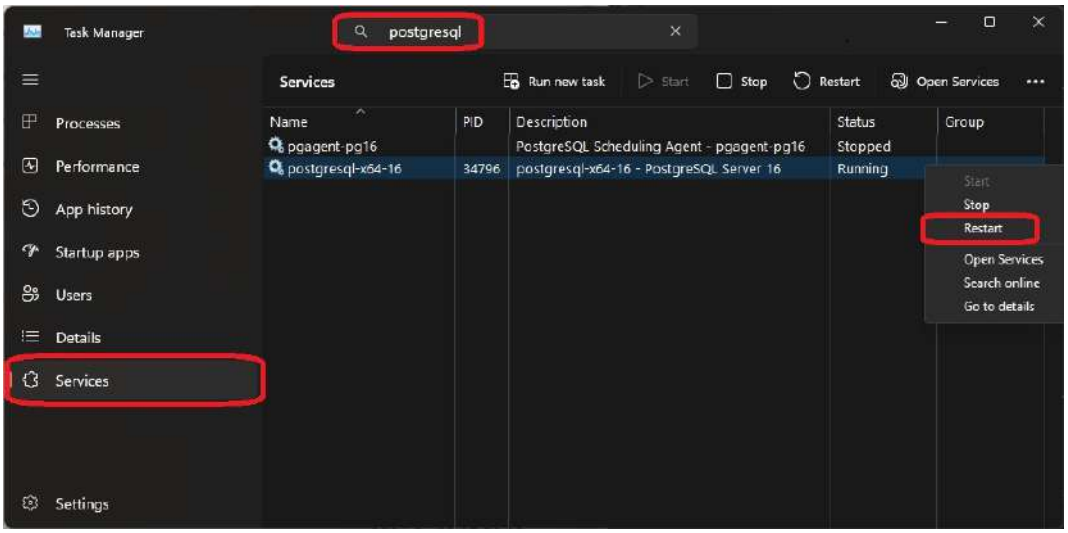
Και το μέγεθος της βάσης δεδομένων μας είναι λίγο μεγαλύτερο από 1GB:



Η PostgreSQL έχει default *shared_buffers* στα 128MB.



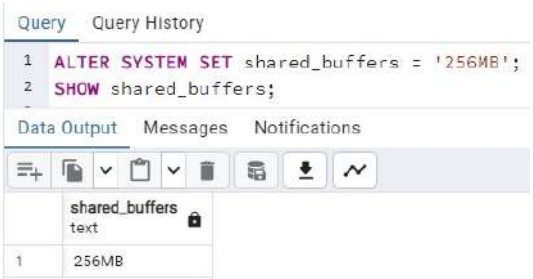
Αλλάζουμε το μέγεθος με την χρήση της εντολή `ALTER SYSTEM SET shared_buffer = <my_value>;` Και στην συνέχεια κάνουμε επανεκκίνηση του server μέσω του Task Manager των Windows, όπως φαίνεται παρακάτω:



Θα εκτελέσουμε τις παρακάτω δοκιμές στην χωρητικότητα του buffer για να διαπιστώσουμε τυχόν διαφορές στους χρόνους εκτέλεσης των queries:

- Αύξηση σε 256MB
- Αύξηση σε 512MB
- Αύξηση σε 1024MB
- Αύξηση σε 2048MB

Αύξηση σε 256MB



```sql
EXPLAIN ANALYZE
SELECT t::date as time_stamp, count(lon) as longtitude, count(lat) as latitude
FROM positions
GROUP BY time_stamp
ORDER BY longtitude desc;
```

Data Output | Messages | Notifications

	QUERY PLAN text
1	Sort (cost=651979.88..653137.63 rows=463100 width=20) (actual time=920.198..924.144 rows=24 loops=1)
2	Sort Key: (count(lon)) DESC
3	Sort Method: quicksort Memory: 26kB
4	-> Finalize GroupAggregate (cost=478101.48..598900.94 rows=463100 width=20) (actual time=920.169..924.133 rows=24 loops=1)
5	Group Key: ((t)::date)
6	-> Gather Merge (cost=478101.48..586165.69 rows=926200 width=20) (actual time=920.163..924.119 rows=72 loops=1)
7	Workers Planned: 2
8	Workers Launched: 2
9	-> Sort (cost=477101.45..478259.20 rows=463100 width=20) (actual time=863.797..863.799 rows=24 loops=3)
10	Sort Key: ((t)::date)
11	Sort Method: quicksort Memory: 26kB
12	Worker 0: Sort Method: quicksort Memory: 26kB
13	Worker 1: Sort Method: quicksort Memory: 26kB
14	-> Partial HashAggregate (cost=383875.11..424022.51 rows=463100 width=20) (actual time=863.573..863.778 rows=24 loops=3)
15	Group Key: (t)::date
16	Planned Partitions: 8 Batches: 1 Memory Usage: 1561kB
17	Worker 0: Batches: 1 Memory Usage: 1561kB
18	Worker 1: Batches: 1 Memory Usage: 1561kB
19	-> Parallel Seq Scan on positions (cost=0.00..165812.22 rows=2931938 width=20) (actual time=0.209..557.216 rows=2345550 loop...
20	Planning Time: 0.116 ms
21	Execution Time: 924.553 ms

```sql
EXPLAIN ANALYZE
SELECT type as vessel_type, count(V.id) as vessels_with_greek_flag
FROM vessels AS V
JOIN positions AS P ON V.id = P.vessel_id
WHERE V.flag = 'Greece'
GROUP BY V.type;
```

Data Output | Messages | Notifications

	QUERY PLAN text
1	Finalize GroupAggregate (cost=174577.63..174585.99 rows=33 width=10) (actual time=1132.177..1135.869 rows=29 loops=1)
2	Group Key: v.type
3	-> Gather Merge (cost=174577.63..174585.33 rows=66 width=10) (actual time=1132.171..1135.852 rows=87 loops=1)
4	Workers Planned: 2
5	Workers Launched: 2
6	-> Sort (cost=173577.60..173577.68 rows=33 width=10) (actual time=1114.564..1114.566 rows=29 loops=3)
7	Sort Key: v.type
8	Sort Method: quicksort Memory: 26kB
9	Worker 0: Sort Method: quicksort Memory: 26kB
10	Worker 1: Sort Method: quicksort Memory: 26kB
11	-> Partial HashAggregate (cost=173576.44..173576.77 rows=33 width=10) (actual time=1114.503..1114.507 rows=29 loops=3)
12	Group Key: v.type
13	Batches: 1 Memory Usage: 24kB
14	Worker 0: Batches: 1 Memory Usage: 24kB
15	Worker 1: Batches: 1 Memory Usage: 24kB
16	-> Hash Join (cost=16.16..166261.58 rows=1462971 width=67) (actual time=0.530..821.345 rows=1824966 loops=3)
17	Hash Cond: ((p.vessel_id)::text = (v.id)::text)
18	-> Parallel Seq Scan on positions p (cost=0.00..158482.38 rows=2931938 width=65) (actual time=0.229..336.784 rows=2345550 loop...
19	-> Hash (cost=13.11..13.11 rows=244 width=67) (actual time=0.286..0.286 rows=244 loops=3)
20	Buckets: 1024 Batches: 1 Memory Usage: 32kB
21	-> Seq Scan on vessels v (cost=0.00..13.11 rows=244 width=67) (actual time=0.171..0.247 rows=244 loops=3)
22	Filter: ((flag)::text = 'Greece'::text)
23	Rows Removed by Filter: 245
24	Planning Time: 0.286 ms
25	Execution Time: 1135.933 ms

```
5  EXPLAIN ANALYZE
6  SELECT V.id, V.type, count(V.type)
7  FROM vessels AS V
8  JOIN positions AS P ON V.id = P.vessel_id
9  WHERE P.speed > 30
10 GROUP BY V.id, V.type;
```

Data Output | Messages | Notifications

	QUERY PLAN text
1	Finalize GroupAggregate (cost=167497.51..167621.40 rows=489 width=75) (actual time=651.354..654.697 rows=18 loops=1)
2	Group Key: v.id
3	-> Gather Merge (cost=167497.51..167611.62 rows=978 width=75) (actual time=651.344..654.683 rows=45 loops=1)
4	Workers Planned: 2
5	Workers Launched: 2
6	-> Sort (cost=166497.49..166498.71 rows=489 width=75) (actual time=634.118..634.120 rows=15 loops=3)
7	Sort Key: v.id
8	Sort Method: quicksort Memory: 26kB
9	Worker 0: Sort Method: quicksort Memory: 26kB
10	Worker 1: Sort Method: quicksort Memory: 26kB
11	-> Partial HashAggregate (cost=166470.75..166475.64 rows=489 width=75) (actual time=634.066..634.071 rows=15 loops=3)
12	Group Key: v.id
13	Batches: 1 Memory Usage: 49kB
14	Worker 0: Batches: 1 Memory Usage: 49kB
15	Worker 1: Batches: 1 Memory Usage: 49kB
16	-> Hash Join (cost=18.00..166051.98 rows=83754 width=67) (actual time=1.178..619.207 rows=69914 loops=3)
17	Hash Cond: ((p.vessel_id)::text = (v.id)::text)
18	-> Parallel Seq Scan on positions p (cost=0.00..165812.22 rows=83754 width=65) (actual time=0.772..600.432 rows=69914 loops=3)
19	Filter: (speed > '30'::numeric)
20	Rows Removed by Filter: 2275637
21	-> Hash (cost=11.89..11.89 rows=489 width=67) (actual time=0.387..0.387 rows=489 loops=3)
22	Buckets: 1024 Batches: 1 Memory Usage: 56kB
23	-> Seq Scan on vessels v (cost=0.00..11.89 rows=489 width=67) (actual time=0.227..0.289 rows=489 loops=3)
24	Planning Time: 0.319 ms
25	Execution Time: 654.758 ms

```
EXPLAIN ANALYZE
SELECT P.t::date AS given_day, count(lon) AS registered_vessel_pisition
FROM positions AS P
JOIN vessels AS V on P.vessel_id = V.id
JOIN vessel_types AS VT ON  V.type = VT.code
WHERE VT.description LIKE 'Passenger,%'
    AND P.t::date BETWEEN '2019-08-14' AND '2019-08-18'
GROUP BY P.t::date;
```

Data Output | Messages | Notifications

	QUERY PLAN text
3	-> Gather Merge (cost=188962.40..189309.32 rows=2766 width=12) (actual time=612.348..628.545 rows=15 loops=1)
4	Workers Planned: 2
5	Workers Launched: 2
6	-> Partial GroupAggregate (cost=187962.37..187990.03 rows=1383 width=12) (actual time=574.099..590.456 rows=5 loops=3)
7	Group Key: ((p.t)::date)
8	-> Sort (cost=187962.37..187965.83 rows=1383 width=12) (actual time=573.058..581.095 rows=120498 loops=3)
9	Sort Key: ((p.t)::date)
10	Sort Method: external merge Disk: 2704kB
11	Worker 0: Sort Method: external merge Disk: 2672kB
12	Worker 1: Sort Method: external merge Disk: 2696kB
13	-> Hash Join (cost=16.24..187890.22 rows=1383 width=12) (actual time=186.911..553.552 rows=120498 loops=3)
14	Hash Cond: ((p.vessel_id)::text = (v.id)::text)
15	-> Parallel Seq Scan on positions p (cost=0.00..187801.76 rows=14660 width=81) (actual time=186.012..495.264 rows=392404 loop...
16	Filter: (((t)::date >= '2019-08-14'::date) AND ((t)::date <= '2019-08-18'::date))
17	Rows Removed by Filter: 1953146
18	-> Hash (cost=15.67..15.67 rows=46 width=65) (actual time=0.673..0.674 rows=64 loops=3)
19	Buckets: 1024 Batches: 1 Memory Usage: 15kB
20	-> Hash Join (cost=2.45..15.67 rows=46 width=65) (actual time=0.571..0.659 rows=64 loops=3)
21	Hash Cond: ((v.type)::text = (vt.code)::text)
22	-> Seq Scan on vessels v (cost=0.00..11.89 rows=489 width=67) (actual time=0.308..0.348 rows=489 loops=3)
23	-> Hash (cost=2.33..2.33 rows=10 width=2) (actual time=0.251..0.252 rows=10 loops=3)
24	Buckets: 1024 Batches: 1 Memory Usage: 9kB
25	-> Seq Scan on vessel_types vt (cost=0.00..2.33 rows=10 width=2) (actual time=0.238..0.242 rows=10 loops=3)
26	Filter: ((description)::text ~~ 'Passenger,%'::text)
27	Rows Removed by Filter: 96
28	Planning Time: 0.405 ms
29	Execution Time: 629.753 ms

```
EXPLAIN ANALYZE
SELECT V.id as vessel_id, P.speed as stationary, V.type as cargo_vessels
FROM positions AS P
JOIN vessels AS V on P.vessel_id = V.id
WHERE V.type BETWEEN '70' and '79'
    AND P.speed = 0.0
    AND P.t >= '2019-08-15 00:00:00' AND P.t <= '2019-08-18 00:00:00'
GROUP BY V.id, P.speed
```

Data Output | Messages | Notifications

	QUERY PLAN text
1	Group (cost=182156.46..182181.49 rows=105 width=71) (actual time=504.395..508.099 rows=30 loops=1)
2	Group Key: v.id
3	-> Gather Merge (cost=182156.46..182180.96 rows=210 width=71) (actual time=504.394..508.085 rows=89 loops=1)
4	Workers Planned: 2
5	Workers Launched: 2
6	-> Sort (cost=181156.44..181156.70 rows=105 width=71) (actual time=485.567..485.569 rows=30 loops=3)
7	Sort Key: v.id
8	Sort Method: quicksort Memory: 27kB
9	Worker 0: Sort Method: quicksort Memory: 27kB
10	Worker 1: Sort Method: quicksort Memory: 27kB
11	-> Partial HashAggregate (cost=181151.86..181152.91 rows=105 width=71) (actual time=485.488..485.494 rows=30 loops=3)
12	Group Key: v.id
13	Batches: 1 Memory Usage: 24kB
14	Worker 0: Batches: 1 Memory Usage: 24kB
15	Worker 1: Batches: 1 Memory Usage: 24kB
16	-> Hash Join (cost=15.65..181039.89 rows=44791 width=71) (actual time=211.608..483.910 rows=7094 loops=3)
17	Hash Cond: ((p.vessel_id)::text = (v.id)::text)
18	-> Parallel Seq Scan on positions p (cost=0.00..180471.91 rows=208600 width=69) (actual time=181.576..466.725 rows=160205 loops=3)
19	Filter: ((t >= '2019-08-15 00:00:00'::timestamp without time zone) AND (t <= '2019-08-18 00:00:00'::timestamp without time zone) AND (s
20	Rows Removed by Filter: 2185346
21	-> Hash (cost=14.34..14.34 rows=105 width=67) (actual time=0.604..0.604 rows=104 loops=3)
22	Buckets: 1024 Batches: 1 Memory Usage: 19kB
23	-> Seq Scan on vessels v (cost=0.00..14.34 rows=105 width=67) (actual time=0.371..0.573 rows=104 loops=3)
24	Filter: (((type)::text >= '70'::text) AND ((type)::text <= '79'::text))
25	Rows Removed by Filter: 385
26	Planning Time: 0.294 ms
27	Execution Time: 508.149 ms

```sql
EXPLAIN ANALYZE
SELECT P.vessel_id as completely_stationary_vessels_between_12_and_18_August2019
FROM positions AS P
JOIN vessels AS V on P.vessel_id = V.id
WHERE V.type::integer BETWEEN 70 and 79
    AND P.t >= '2019-08-12 00:00:00' AND P.t <= '2019-08-19 00:00:00'
GROUP BY P.vessel_id
HAVING MAX(P.speed) = 0.0;
```

Data Output | Messages | Notifications

	QUERY PLAN text
1	Finalize GroupAggregate (cost=175942.68..176072.80 rows=2 width=65) (actual time=520.943..523.065 rows=1 loops=1)
2	Group Key: p.vessel_id
3	Filter: (max(p.speed) = 0.0)
4	Rows Removed by Filter: 48
5	-> Gather Merge (cost=175942.68..176063.30 rows=844 width=97) (actual time=516.371..523.023 rows=143 loops=1)
6	Workers Planned: 2
7	Workers Launched: 2
8	-> Partial GroupAggregate (cost=174942.66..174965.86 rows=422 width=97) (actual time=490.096..495.972 rows=48 loops=3)
9	Group Key: p.vessel_id
10	-> Sort (cost=174942.66..174948.99 rows=2531 width=69) (actual time=490.030..491.329 rows=29614 loops=3)
11	Sort Key: p.vessel_id
12	Sort Method: quicksort Memory: 3576kB
13	Worker 0: Sort Method: quicksort Memory: 3509kB
14	Worker 1: Sort Method: quicksort Memory: 3550kB
15	-> Hash Join (cost=19.25..174799.59 rows=2531 width=69) (actual time=37.135..458.469 rows=29614 loops=3)
16	Hash Cond: ((p.vessel_id)::text = (v.id)::text)
17	-> Parallel Seq Scan on positions p (cost=0.00..173142.07 rows=618741 width=69) (actual time=31.871..411.578 rows=48868...
18	Filter: ((t >= '2019-08-12 00:00:00'::timestamp without time zone) AND (t <= '2019-08-19 00:00:00'::timestamp without time z...
19	Rows Removed by Filter: 1856870
20	-> Hash (cost=19.23..19.23 rows=2 width=65) (actual time=0.449..0.450 rows=104 loops=3)
21	Buckets: 1024 Batches: 1 Memory Usage: 18kB
22	-> Seq Scan on vessels v (cost=0.00..19.23 rows=2 width=65) (actual time=0.351..0.425 rows=104 loops=3)
23	Filter: (((type)::integer >= 70) AND ((type)::integer <= 79))
24	Rows Removed by Filter: 385
25	Planning Time: 0.290 ms
26	Execution Time: 523.458 ms

Αύξηση σε 512MB

```
SHOW shared_buffers
```

Data Output | Messages | N

	shared_buffers text
1	512MB

```
EXPLAIN ANALYZE
SELECT t::date as time_stamp, count(lon) as longtitude, count(lat) as latitude
FROM positions
GROUP BY time_stamp
ORDER BY longtitude desc;
```

Data Output | Messages | Notifications

	QUERY PLAN text
1	Sort (cost=651979.88..653137.63 rows=463100 width=20) (actual time=939.688..943.609 rows=24 loops=1)
2	Sort Key: (count(lon)) DESC
3	Sort Method: quicksort Memory: 26kB
4	-> Finalize GroupAggregate (cost=478101.48..598900.94 rows=463100 width=20) (actual time=939.663..943.602 rows=24 loops=1)
5	Group Key: ((t)::date)
6	-> Gather Merge (cost=478101.48..586165.69 rows=926200 width=20) (actual time=939.658..943.588 rows=72 loops=1)
7	Workers Planned: 2
8	Workers Launched: 2
9	-> Sort (cost=477101.45..478259.20 rows=463100 width=20) (actual time=853.020..853.021 rows=24 loops=3)
10	Sort Key: ((t)::date)
11	Sort Method: quicksort Memory: 26kB
12	Worker 0: Sort Method: quicksort Memory: 26kB
13	Worker 1: Sort Method: quicksort Memory: 26kB
14	-> Partial HashAggregate (cost=383875.11..424022.51 rows=463100 width=20) (actual time=852.934..853.002 rows=24 loops=3)
15	Group Key: (t)::date
16	Planned Partitions: 8 Batches: 1 Memory Usage: 1561kB
17	Worker 0: Batches: 1 Memory Usage: 1561kB
18	Worker 1: Batches: 1 Memory Usage: 1561kB
19	-> Parallel Seq Scan on positions (cost=0.00..165812.22 rows=2931938 width=20) (actual time=0.209..552.163 rows=2345550 loop...
20	Planning Time: 0.105 ms
21	Execution Time: 943.935 ms

```
EXPLAIN ANALYZE
SELECT type as vessel_type, count(V.id) as vessels_with_greek_flag
FROM vessels AS V
JOIN positions AS P ON V.id = P.vessel_id
WHERE V.flag = 'Greece'
GROUP BY V.type;
```

Data Output | Messages | Notifications

	QUERY PLAN text
1	Finalize GroupAggregate (cost=174577.63..174585.99 rows=33 width=10) (actual time=1159.884..1163.514 rows=29 loops=1)
2	Group Key: v.type
3	-> Gather Merge (cost=174577.63..174585.33 rows=66 width=10) (actual time=1159.876..1163.494 rows=87 loops=1)
4	Workers Planned: 2
5	Workers Launched: 2
6	-> Sort (cost=173577.60..173577.68 rows=33 width=10) (actual time=1138.079..1138.082 rows=29 loops=3)
7	Sort Key: v.type
8	Sort Method: quicksort Memory: 26kB
9	Worker 0: Sort Method: quicksort Memory: 26kB
10	Worker 1: Sort Method: quicksort Memory: 26kB
11	-> Partial HashAggregate (cost=173576.44..173576.77 rows=33 width=10) (actual time=1138.010..1138.014 rows=29 loops=3)
12	Group Key: v.type
13	Batches: 1 Memory Usage: 24kB
14	Worker 0: Batches: 1 Memory Usage: 24kB
15	Worker 1: Batches: 1 Memory Usage: 24kB
16	-> Hash Join (cost=16.16..166261.58 rows=1462971 width=67) (actual time=0.509..844.135 rows=1824966 loops=3)
17	Hash Cond: ((p.vessel_id)::text = (v.id)::text)
18	-> Parallel Seq Scan on positions p (cost=0.00..158482.38 rows=2931938 width=65) (actual time=0.192..354.386 rows=2345550 loop...
19	-> Hash (cost=13.11..13.11 rows=244 width=67) (actual time=0.297..0.297 rows=244 loops=3)
20	Buckets: 1024 Batches: 1 Memory Usage: 32kB
21	-> Seq Scan on vessels v (cost=0.00..13.11 rows=244 width=67) (actual time=0.193..0.253 rows=244 loops=3)
22	Filter: ((flag)::text = 'Greece'::text)
23	Rows Removed by Filter: 245
24	Planning Time: 0.288 ms
25	Execution Time: 1163.556 ms

```sql
EXPLAIN ANALYZE
SELECT V.id, V.type, count(V.type)
FROM vessels AS V
JOIN positions AS P ON V.id = P.vessel_id
WHERE P.speed > 30
GROUP BY V.id, V.type;
```

Data Output | Messages | Notifications

	QUERY PLAN text
1	Finalize GroupAggregate (cost=167497.51..167621.40 rows=489 width=75) (actual time=614.067..617.803 rows=18 loops=1)
2	Group Key: v.id
3	-> Gather Merge (cost=167497.51..167611.62 rows=978 width=75) (actual time=614.062..617.791 rows=45 loops=1)
4	Workers Planned: 2
5	Workers Launched: 2
6	-> Sort (cost=166497.49..166498.71 rows=489 width=75) (actual time=596.720..596.722 rows=15 loops=3)
7	Sort Key: v.id
8	Sort Method: quicksort Memory: 26kB
9	Worker 0: Sort Method: quicksort Memory: 26kB
10	Worker 1: Sort Method: quicksort Memory: 26kB
11	-> Partial HashAggregate (cost=166470.75..166475.64 rows=489 width=75) (actual time=596.666..596.671 rows=15 loops=3)
12	Group Key: v.id
13	Batches: 1 Memory Usage: 49kB
14	Worker 0: Batches: 1 Memory Usage: 49kB
15	Worker 1: Batches: 1 Memory Usage: 49kB
16	-> Hash Join (cost=18.00..166051.98 rows=83754 width=67) (actual time=1.341..582.521 rows=69914 loops=3)
17	Hash Cond: ((p.vessel_id)::text = (v.id)::text)
18	-> Parallel Seq Scan on positions p (cost=0.00..165812.22 rows=83754 width=65) (actual time=0.927..564.796 rows=69914 loop...
19	Filter: (speed > '30'::numeric)
20	Rows Removed by Filter: 2275637
21	-> Hash (cost=11.89..11.89 rows=489 width=67) (actual time=0.387..0.387 rows=489 loops=3)
22	Buckets: 1024 Batches: 1 Memory Usage: 56kB
23	-> Seq Scan on vessels v (cost=0.00..11.89 rows=489 width=67) (actual time=0.240..0.300 rows=489 loops=3)
24	Planning Time: 0.306 ms
25	Execution Time: 617.878 ms

```sql
EXPLAIN ANALYZE
SELECT P.t::date AS given_day, count(lon) AS registered_vessel_pisition
FROM positions AS P
JOIN vessels AS V on P.vessel_id = V.id
JOIN vessel_types AS VT ON  V.type = VT.code
WHERE VT.description LIKE 'Passenger,%'
    AND P.t::date BETWEEN '2019-08-14' AND '2019-08-18'
GROUP BY P.t::date;
```

Data Output | Messages | Notifications

	QUERY PLAN text
1	Finalize GroupAggregate (cost=188962.40..189364.64 rows=3319 width=12) (actual time=606.235..618.859 rows=5 loops=1)
2	Group Key: ((p.t)::date)
3	-> Gather Merge (cost=188962.40..189309.32 rows=2766 width=12) (actual time=602.818..618.844 rows=15 loops=1)
4	Workers Planned: 2
5	Workers Launched: 2
6	-> Partial GroupAggregate (cost=187962.37..187990.03 rows=1383 width=12) (actual time=565.093..581.596 rows=5 loops=3)
7	Group Key: ((p.t)::date)
8	-> Sort (cost=187962.37..187965.83 rows=1383 width=12) (actual time=564.063..572.085 rows=120498 loops=3)
9	Sort Key: ((p.t)::date)
10	Sort Method: external merge Disk: 2656kB
11	Worker 0: Sort Method: external merge Disk: 2728kB
12	Worker 1: Sort Method: external merge Disk: 2688kB
13	-> Hash Join (cost=16.24..187890.22 rows=1383 width=12) (actual time=188.695..545.157 rows=120498 loops=3)
14	Hash Cond: ((p.vessel_id)::text = (v.id)::text)
15	-> Parallel Seq Scan on positions p (cost=0.00..187801.76 rows=14660 width=81) (actual time=187.970..489.391 rows=392404 loop...
16	Filter: ((((t)::date >= '2019-08-14'::date) AND ((t)::date <= '2019-08-18'::date))
17	Rows Removed by Filter: 1953146
18	-> Hash (cost=15.67..15.67 rows=46 width=65) (actual time=0.654..0.656 rows=64 loops=3)
19	Buckets: 1024 Batches: 1 Memory Usage: 15kB
20	-> Hash Join (cost=2.45..15.67 rows=46 width=65) (actual time=0.561..0.643 rows=64 loops=3)
21	Hash Cond: ((v.type)::text = (vt.code)::text)
22	-> Seq Scan on vessels v (cost=0.00..11.89 rows=489 width=67) (actual time=0.348..0.385 rows=489 loops=3)
23	-> Hash (cost=2.33..2.33 rows=10 width=2) (actual time=0.201..0.201 rows=10 loops=3)
24	Buckets: 1024 Batches: 1 Memory Usage: 9kB
25	-> Seq Scan on vessel_types vt (cost=0.00..2.33 rows=10 width=2) (actual time=0.186..0.191 rows=10 loops=3)
26	Filter: ((description)::text ~~ 'Passenger,%'::text)
27	Rows Removed by Filter: 96
28	Planning Time: 0.408 ms
29	Execution Time: 619.989 ms

```
5  EXPLAIN ANALYZE
6  SELECT V.id as vessel_id, P.speed as stationary, V.type as cargo_vessels
7  FROM positions AS P
8  JOIN vessels AS V on P.vessel_id = V.id
9  WHERE V.type BETWEEN '70' and '79'
10     AND P.speed = 0.0
11     AND P.t >= '2019-08-15 00:00:00' AND P.t <= '2019-08-18 00:00:00'
12 GROUP BY V.id, P.speed;
```

Data Output | Messages | Notifications

	QUERY PLAN text
1	Group (cost=182156.46..182181.49 rows=105 width=71) (actual time=543.487..547.055 rows=30 loops=1)
2	Group Key: v.id
3	-> Gather Merge (cost=182156.46..182180.96 rows=210 width=71) (actual time=543.485..547.042 rows=89 loops=1)
4	Workers Planned: 2
5	Workers Launched: 2
6	-> Sort (cost=181156.44..181156.70 rows=105 width=71) (actual time=523.525..523.528 rows=30 loops=3)
7	Sort Key: v.id
8	Sort Method: quicksort Memory: 27kB
9	Worker 0: Sort Method: quicksort Memory: 27kB
10	Worker 1: Sort Method: quicksort Memory: 27kB
11	-> Partial HashAggregate (cost=181151.86..181152.91 rows=105 width=71) (actual time=523.446..523.452 rows=30 loops=3)
12	Group Key: v.id
13	Batches: 1 Memory Usage: 24kB
14	Worker 0: Batches: 1 Memory Usage: 24kB
15	Worker 1: Batches: 1 Memory Usage: 24kB
16	-> Hash Join (cost=15.65..181039.89 rows=44791 width=71) (actual time=206.213..521.982 rows=7094 loops=3)
17	Hash Cond: ((p.vessel_id)::text = (v.id)::text)
18	-> Parallel Seq Scan on positions p (cost=0.00..180471.91 rows=208600 width=69) (actual time=176.976..505.650 rows=160205 loops=3)
19	Filter: ((t >= '2019-08-15 00:00:00'::timestamp without time zone) AND (t <= '2019-08-18 00:00:00'::timestamp without time zone) AND (speed =...
20	Rows Removed by Filter: 2185346
21	-> Hash (cost=14.34..14.34 rows=105 width=67) (actual time=0.521..0.522 rows=104 loops=3)
22	Buckets: 1024 Batches: 1 Memory Usage: 19kB
23	-> Seq Scan on vessels v (cost=0.00..14.34 rows=105 width=67) (actual time=0.332..0.496 rows=104 loops=3)
24	Filter: (((type)::text >= '70'::text) AND ((type)::text <= '79'::text))
25	Rows Removed by Filter: 385
26	Planning Time: 0.341 ms
27	Execution Time: 547.125 ms

```
6  EXPLAIN ANALYZE
7  SELECT P.vessel_id as completely_stationary_vessels_between_12_and_18_August2019
8  FROM positions AS P
9  JOIN vessels AS V on P.vessel_id = V.id
10 WHERE V.type::integer BETWEEN 70 and 79
11     AND P.t >= '2019-08-12 00:00:00' AND P.t <= '2019-08-19 00:00:00'
12 GROUP BY P.vessel_id
13 HAVING MAX(P.speed) = 0.0;
```

Data Output | Messages | Notifications

	QUERY PLAN text
1	Finalize GroupAggregate (cost=175942.68..176072.80 rows=2 width=65) (actual time=530.670..532.725 rows=1 loops=1)
2	Group Key: p.vessel_id
3	Filter: (max(p.speed) = 0.0)
4	Rows Removed by Filter: 48
5	-> Gather Merge (cost=175942.68..176063.30 rows=844 width=97) (actual time=526.074..532.679 rows=144 loops=1)
6	Workers Planned: 2
7	Workers Launched: 2
8	-> Partial GroupAggregate (cost=174942.66..174965.86 rows=422 width=97) (actual time=499.439..505.570 rows=48 loops=3)
9	Group Key: p.vessel_id
10	-> Sort (cost=174942.66..174948.99 rows=2531 width=69) (actual time=499.360..500.949 rows=29614 loops=3)
11	Sort Key: p.vessel_id
12	Sort Method: quicksort Memory: 3517kB
13	Worker 0: Sort Method: quicksort Memory: 3633kB
14	Worker 1: Sort Method: quicksort Memory: 3485kB
15	-> Hash Join (cost=19.25..174799.59 rows=2531 width=69) (actual time=40.409..467.028 rows=29614 loops=3)
16	Hash Cond: ((p.vessel_id)::text = (v.id)::text)
17	-> Parallel Seq Scan on positions p (cost=0.00..173142.07 rows=618741 width=69) (actual time=35.320..419.822 rows=48868...
18	Filter: ((t >= '2019-08-12 00:00:00'::timestamp without time zone) AND (t <= '2019-08-19 00:00:00'::timestamp without time z...
19	Rows Removed by Filter: 1856870
20	-> Hash (cost=19.23..19.23 rows=2 width=65) (actual time=0.454..0.455 rows=104 loops=3)
21	Buckets: 1024 Batches: 1 Memory Usage: 18kB
22	-> Seq Scan on vessels v (cost=0.00..19.23 rows=2 width=65) (actual time=0.359..0.430 rows=104 loops=3)
23	Filter: (((type)::integer >= 70) AND ((type)::integer <= 79))
24	Rows Removed by Filter: 385
25	Planning Time: 0.325 ms
26	Execution Time: 532.998 ms

Αύξηση σε 1024MB

```
2  SHOW shared_buffers;
```

Data Output | Messages | Noti

	shared_buffers text
1	1GB

```
6  EXPLAIN ANALYZE
7  SELECT t::date as time_stamp, count(lon) as longtitude, count(lat) as latitude
8  FROM positions
9  GROUP BY time_stamp
10 ORDER BY longtitude desc;
```

Data Output | Messages | Notifications

	QUERY PLAN text
1	Sort (cost=651979.88..658137.63 rows=463100 width=20) (actual time=890.975..894.421 rows=24 loops=1)
2	Sort Key: (count(lon)) DESC
3	Sort Method: quicksort Memory: 26kB
4	-> Finalize GroupAggregate (cost=478101.48..598900.94 rows=463100 width=20) (actual time=890.949..894.414 rows=24 loops=1)
5	Group Key: ((t)::date)
6	-> Gather Merge (cost=478101.48..586165.69 rows=926200 width=20) (actual time=890.945..894.400 rows=72 loops=1)
7	Workers Planned: 2
8	Workers Launched: 2
9	-> Sort (cost=477101.45..478259.20 rows=463100 width=20) (actual time=835.644..835.645 rows=24 loops=3)
10	Sort Key: ((t)::date)
11	Sort Method: quicksort Memory: 26kB
12	Worker 0: Sort Method: quicksort Memory: 26kB
13	Worker 1: Sort Method: quicksort Memory: 26kB
14	-> Partial HashAggregate (cost=383875.11..424022.51 rows=463100 width=20) (actual time=835.493..835.623 rows=24 loops=3)
15	Group Key: (t)::date
16	Planned Partitions: 8 Batches: 1 Memory Usage: 1561kB
17	Worker 0: Batches: 1 Memory Usage: 1561kB
18	Worker 1: Batches: 1 Memory Usage: 1561kB
19	-> Parallel Seq Scan on positions (cost=0.00..165812.22 rows=2931938 width=20) (actual time=0.177..537.446 rows=2345550 loop...
20	Planning Time: 0.091 ms
21	Execution Time: 894.718 ms

```
6  EXPLAIN ANALYZE
7  SELECT type as vessel_type, count(V.id) as vessels_with_greek_flag
8  FROM vessels AS V
9  JOIN positions AS P ON V.id = P.vessel_id
10 WHERE V.flag = 'Greece'
11 GROUP BY V.type;
```

Data Output | Messages | Notifications

	QUERY PLAN text
1	Finalize GroupAggregate (cost=174577.63..174585.99 rows=83 width=10) (actual time=1220.977..1225.469 rows=29 loops=1)
2	Group Key: v.type
3	-> Gather Merge (cost=174577.63..174585.33 rows=66 width=10) (actual time=1220.963..1225.445 rows=87 loops=1)
4	Workers Planned: 2
5	Workers Launched: 2
6	-> Sort (cost=173577.60..173577.68 rows=33 width=10) (actual time=1199.796..1199.799 rows=29 loops=3)
7	Sort Key: v.type
8	Sort Method: quicksort Memory: 26kB
9	Worker 0: Sort Method: quicksort Memory: 26kB
10	Worker 1: Sort Method: quicksort Memory: 26kB
11	-> Partial HashAggregate (cost=173576.44..173576.77 rows=33 width=10) (actual time=1199.730..1199.734 rows=29 loops=3)
12	Group Key: v.type
13	Batches: 1 Memory Usage: 24kB
14	Worker 0: Batches: 1 Memory Usage: 24kB
15	Worker 1: Batches: 1 Memory Usage: 24kB
16	-> Hash Join (cost=16.16..166261.58 rows=1462971 width=67) (actual time=0.506..892.390 rows=1824966 loops=3)
17	Hash Cond: ((p.vessel_id)::text = (v.id)::text)
18	-> Parallel Seq Scan on positions p (cost=0.00..158482.38 rows=2931938 width=65) (actual time=0.227..378.309 rows=2345550 loop...
19	-> Hash (cost=13.11..13.11 rows=244 width=67) (actual time=0.259..0.259 rows=244 loops=3)
20	Buckets: 1024 Batches: 1 Memory Usage: 32kB
21	-> Seq Scan on vessels v (cost=0.00..13.11 rows=244 width=67) (actual time=0.154..0.219 rows=244 loops=3)
22	Filter: ((flag)::text = 'Greece'::text)
23	Rows Removed by Filter: 245
24	Planning Time: 0.442 ms
25	Execution Time: 1225.562 ms

```
EXPLAIN ANALYZE
SELECT V.id, V.type, count(V.type)
FROM vessels AS V
JOIN positions AS P ON V.id = P.vessel_id
WHERE P.speed > 30
GROUP BY V.id, V.type;
```

Data Output | Messages | Notifications

	QUERY PLAN text
1	Finalize GroupAggregate (cost=167497.51..167621.40 rows=489 width=75) (actual time=662.355..665.986 rows=18 loops=1)
2	Group Key: v.id
3	-> Gather Merge (cost=167497.51..167611.62 rows=978 width=75) (actual time=662.349..665.973 rows=45 loops=1)
4	Workers Planned: 2
5	Workers Launched: 2
6	-> Sort (cost=166497.49..166498.71 rows=489 width=75) (actual time=641.571..641.573 rows=15 loops=3)
7	Sort Key: v.id
8	Sort Method: quicksort Memory: 26kB
9	Worker 0: Sort Method: quicksort Memory: 26kB
10	Worker 1: Sort Method: quicksort Memory: 26kB
11	-> Partial HashAggregate (cost=166470.75..166475.64 rows=489 width=75) (actual time=641.523..641.528 rows=15 loops=3)
12	Group Key: v.id
13	Batches: 1 Memory Usage: 49kB
14	Worker 0: Batches: 1 Memory Usage: 49kB
15	Worker 1: Batches: 1 Memory Usage: 49kB
16	-> Hash Join (cost=18.00..166051.98 rows=83754 width=67) (actual time=0.708..627.046 rows=69914 loops=3)
17	Hash Cond: ((p.vessel_id)::text = (v.id)::text)
18	-> Parallel Seq Scan on positions p (cost=0.00..165812.22 rows=83754 width=65) (actual time=0.399..609.058 rows=69914 loop...
19	Filter: (speed > '30'::numeric)
20	Rows Removed by Filter: 2275637
21	-> Hash (cost=11.89..11.89 rows=489 width=67) (actual time=0.292..0.292 rows=489 loops=3)
22	Buckets: 1024 Batches: 1 Memory Usage: 56kB
23	-> Seq Scan on vessels v (cost=0.00..11.89 rows=489 width=67) (actual time=0.153..0.210 rows=489 loops=3)
24	Planning Time: 0.329 ms
25	Execution Time: 666.066 ms

```
EXPLAIN ANALYZE
SELECT P.t::date AS given_day, count(lon) AS registered_vessel_pisition
FROM positions AS P
JOIN vessels AS V on P.vessel_id = V.id
JOIN vessel_types AS VT ON  V.type = VT.code
WHERE VT.description LIKE 'Passenger,%'
    AND P.t::date BETWEEN '2019-08-14' AND '2019-08-18'
GROUP BY P.t::date;
```

Data Output | Messages | Notifications

	QUERY PLAN text
1	Finalize GroupAggregate (cost=188962.40..189364.64 rows=8319 width=12) (actual time=705.856..719.063 rows=5 loops=1)
2	Group Key: ((p.t)::date)
3	-> Gather Merge (cost=188962.40..189309.32 rows=2766 width=12) (actual time=702.331..719.047 rows=15 loops=1)
4	Workers Planned: 2
5	Workers Launched: 2
6	-> Partial GroupAggregate (cost=187962.37..187990.03 rows=1383 width=12) (actual time=660.910..679.826 rows=5 loops=3)
7	Group Key: ((p.t)::date)
8	-> Sort (cost=187962.37..187965.83 rows=1383 width=12) (actual time=659.282..668.392 rows=120498 loops=3)
9	Sort Key: ((p.t)::date)
10	Sort Method: external merge Disk: 2672kB
11	Worker 0: Sort Method: external merge Disk: 2688kB
12	Worker 1: Sort Method: external merge Disk: 2712kB
13	-> Hash Join (cost=16.24..187890.22 rows=1383 width=12) (actual time=214.786..636.454 rows=120498 loops=3)
14	Hash Cond: ((p.vessel_id)::text = (v.id)::text)
15	-> Parallel Seq Scan on positions p (cost=0.00..187801.76 rows=14660 width=81) (actual time=214.074..577.480 rows=392404 loop...
16	Filter: (((t)::date >= '2019-08-14'::date) AND ((t)::date <= '2019-08-18'::date))
17	Rows Removed by Filter: 1953146
18	-> Hash (cost=15.67..15.67 rows=46 width=65) (actual time=0.648..0.650 rows=64 loops=3)
19	Buckets: 1024 Batches: 1 Memory Usage: 15kB
20	-> Hash Join (cost=2.45..15.67 rows=46 width=65) (actual time=0.546..0.635 rows=64 loops=3)
21	Hash Cond: ((v.type)::text = (vt.code)::text)
22	-> Seq Scan on vessels v (cost=0.00..11.89 rows=489 width=67) (actual time=0.332..0.375 rows=489 loops=3)
23	-> Hash (cost=2.33..2.33 rows=10 width=2) (actual time=0.199..0.200 rows=10 loops=3)
24	Buckets: 1024 Batches: 1 Memory Usage: 9kB
25	-> Seq Scan on vessel_types vt (cost=0.00..2.33 rows=10 width=2) (actual time=0.181..0.186 rows=10 loops=3)
26	Filter: ((description)::text ~~ 'Passenger,%'::text)
27	Rows Removed by Filter: 96
28	Planning Time: 0.445 ms
29	Execution Time: 720.286 ms

```
EXPLAIN ANALYZE
SELECT V.id as vessel_id, P.speed as stationary, V.type as cargo_vessels
FROM positions AS P
JOIN vessels AS V on P.vessel_id = V.id
WHERE V.type BETWEEN '70' and '79'
    AND P.speed = 0.0
    AND P.t >= '2019-08-15 00:00:00' AND P.t <= '2019-08-18 00:00:00'
GROUP BY V.id, P.speed
```

Data Output | Messages | Notifications

	QUERY PLAN text
1	Group (cost=182156.46..182181.49 rows=105 width=71) (actual time=507.275..510.736 rows=30 loops=1)
2	Group Key: v.id
3	-> Gather Merge (cost=182156.46..182180.96 rows=210 width=71) (actual time=507.274..510.723 rows=89 loops=1)
4	Workers Planned: 2
5	Workers Launched: 2
6	-> Sort (cost=181156.44..181156.70 rows=105 width=71) (actual time=489.075..489.078 rows=30 loops=3)
7	Sort Key: v.id
8	Sort Method: quicksort Memory: 27kB
9	Worker 0: Sort Method: quicksort Memory: 27kB
10	Worker 1: Sort Method: quicksort Memory: 27kB
11	-> Partial HashAggregate (cost=181151.86..181152.91 rows=105 width=71) (actual time=488.996..489.005 rows=30 loops=3)
12	Group Key: v.id
13	Batches: 1 Memory Usage: 24kB
14	Worker 0: Batches: 1 Memory Usage: 24kB
15	Worker 1: Batches: 1 Memory Usage: 24kB
16	-> Hash Join (cost=15.65..181039.89 rows=44791 width=71) (actual time=210.106..487.484 rows=7094 loops=3)
17	Hash Cond: ((p.vessel_id)::text = (v.id)::text)
18	-> Parallel Seq Scan on positions p (cost=0.00..180471.91 rows=208600 width=69) (actual time=181.165..471.408 rows=160205 loops=3)
19	Filter: ((t >= '2019-08-15 00:00:00'::timestamp without time zone) AND (t <= '2019-08-18 00:00:00'::timestamp without time zone) AND (s
20	Rows Removed by Filter: 2185846
21	-> Hash (cost=14.34..14.34 rows=105 width=67) (actual time=0.513..0.513 rows=104 loops=3)
22	Buckets: 1024 Batches: 1 Memory Usage: 19kB
23	-> Seq Scan on vessels v (cost=0.00..14.34 rows=105 width=67) (actual time=0.327..0.491 rows=104 loops=3)
24	Filter: (((type)::text >= '70'::text) AND ((type)::text <= '79'::text))
25	Rows Removed by Filter: 385
26	Planning Time: 0.307 ms
27	Execution Time: 510.772 ms

```
EXPLAIN ANALYZE
SELECT P.vessel_id as completely_stationary_vessels_between_12_and_18_August2019
FROM positions AS P
JOIN vessels AS V on P.vessel_id = V.id
WHERE V.type::integer BETWEEN 70 and 79
    AND P.t >= '2019-08-12 00:00:00' AND P.t <= '2019-08-19 00:00:00'
GROUP BY P.vessel_id
HAVING MAX(P.speed) = 0.0;
```

Data Output | Messages | Notifications

	QUERY PLAN text
1	Finalize GroupAggregate (cost=175942.68..176072.80 rows=2 width=65) (actual time=525.896..527.789 rows=1 loops=1)
2	Group Key: p.vessel_id
3	Filter: (max(p.speed) = 0.0)
4	Rows Removed by Filter: 48
5	-> Gather Merge (cost=175942.68..176063.30 rows=844 width=97) (actual time=521.401..527.744 rows=144 loops=1)
6	Workers Planned: 2
7	Workers Launched: 2
8	-> Partial GroupAggregate (cost=174942.66..174965.86 rows=422 width=97) (actual time=497.736..504.720 rows=48 loops=3)
9	Group Key: p.vessel_id
10	-> Sort (cost=174942.66..174948.99 rows=2531 width=69) (actual time=497.656..499.781 rows=29614 loops=3)
11	Sort Key: p.vessel_id
12	Sort Method: quicksort Memory: 3565kB
13	Worker 0: Sort Method: quicksort Memory: 3523kB
14	Worker 1: Sort Method: quicksort Memory: 3547kB
15	-> Hash Join (cost=19.25..174799.59 rows=2531 width=69) (actual time=34.837..466.010 rows=29614 loops=3)
16	Hash Cond: ((p.vessel_id)::text = (v.id)::text)
17	-> Parallel Seq Scan on positions p (cost=0.00..173142.07 rows=618741 width=69) (actual time=29.288..419.282 rows=48868...
18	Filter: ((t >= '2019-08-12 00:00:00'::timestamp without time zone) AND (t <= '2019-08-19 00:00:00'::timestamp without time z...
19	Rows Removed by Filter: 1856870
20	-> Hash (cost=19.23..19.23 rows=2 width=65) (actual time=0.465..0.466 rows=104 loops=3)
21	Buckets: 1024 Batches: 1 Memory Usage: 18kB
22	-> Seq Scan on vessels v (cost=0.00..19.23 rows=2 width=65) (actual time=0.373..0.445 rows=104 loops=3)
23	Filter: (((type)::integer >= 70) AND ((type)::integer <= 79))
24	Rows Removed by Filter: 385
25	Planning Time: 0.310 ms
26	Execution Time: 528.128 ms

Αύξηση σε 2048MB

```
SHOW shared_buffers;
```

Data Output | Messages | Not

	shared_buffers text
1	2GB

```
EXPLAIN ANALYZE
SELECT t::date as time_stamp, count(lon) as longtitude, count(lat) as latitude
FROM positions
GROUP BY time_stamp
ORDER BY longtitude desc;
```

Data Output | Messages | Notifications

	QUERY PLAN text
1	Sort (cost=651979.88..653137.63 rows=463100 width=20) (actual time=846.015..850.348 rows=24 loops=1)
2	Sort Key: (count(lon)) DESC
3	Sort Method: quicksort Memory: 26kB
4	-> Finalize GroupAggregate (cost=478101.48..598900.94 rows=463100 width=20) (actual time=845.987..850.339 rows=24 loops=1)
5	Group Key: ((t)::date)
6	-> Gather Merge (cost=478101.48..586165.69 rows=926200 width=20) (actual time=845.980..850.323 rows=72 loops=1)
7	Workers Planned: 2
8	Workers Launched: 2
9	-> Sort (cost=477101.45..478259.20 rows=463100 width=20) (actual time=828.686..828.687 rows=24 loops=3)
10	Sort Key: ((t)::date)
11	Sort Method: quicksort Memory: 26kB
12	Worker 0: Sort Method: quicksort Memory: 26kB
13	Worker 1: Sort Method: quicksort Memory: 26kB
14	-> Partial HashAggregate (cost=383875.11..424022.51 rows=463100 width=20) (actual time=828.489..828.668 rows=24 loops=3)
15	Group Key: (t)::date
16	Planned Partitions: 8 Batches: 1 Memory Usage: 1561kB
17	Worker 0: Batches: 1 Memory Usage: 1561kB
18	Worker 1: Batches: 1 Memory Usage: 1561kB
19	-> Parallel Seq Scan on positions (cost=0.00..165812.22 rows=2931938 width=20) (actual time=0.216..536.465 rows=2345550 loop...
20	Planning Time: 0.088 ms
21	Execution Time: 850.713 ms

```
EXPLAIN ANALYZE
SELECT type as vessel_type, count(V.id) as vessels_with_greek_flag
FROM vessels AS V
JOIN positions AS P ON V.id = P.vessel_id
WHERE V.flag = 'Greece'
GROUP BY V.type;
```

Data Output | Messages | Notifications

	QUERY PLAN text
1	Finalize GroupAggregate (cost=174577.63..174585.99 rows=33 width=10) (actual time=1134.052..1138.995 rows=29 loops=1)
2	Group Key: v.type
3	-> Gather Merge (cost=174577.63..174585.33 rows=66 width=10) (actual time=1134.042..1138.971 rows=87 loops=1)
4	Workers Planned: 2
5	Workers Launched: 2
6	-> Sort (cost=173577.60..173577.68 rows=33 width=10) (actual time=1114.453..1114.456 rows=29 loops=3)
7	Sort Key: v.type
8	Sort Method: quicksort Memory: 26kB
9	Worker 0: Sort Method: quicksort Memory: 26kB
10	Worker 1: Sort Method: quicksort Memory: 26kB
11	-> Partial HashAggregate (cost=173576.44..173576.77 rows=33 width=10) (actual time=1114.371..1114.377 rows=29 loops=3)
12	Group Key: v.type
13	Batches: 1 Memory Usage: 24kB
14	Worker 0: Batches: 1 Memory Usage: 24kB
15	Worker 1: Batches: 1 Memory Usage: 24kB
16	-> Hash Join (cost=16.16..166261.58 rows=1462971 width=67) (actual time=0.490..823.722 rows=1824966 loops=3)
17	Hash Cond: ((p.vessel_id)::text = (v.id)::text)
18	-> Parallel Seq Scan on positions p (cost=0.00..156482.38 rows=2931938 width=65) (actual time=0.190..337.371 rows=2345550 loop...
19	-> Hash (cost=13.11..13.11 rows=244 width=67) (actual time=0.281..0.281 rows=244 loops=3)
20	Buckets: 1024 Batches: 1 Memory Usage: 32kB
21	-> Seq Scan on vessels v (cost=0.00..13.11 rows=244 width=67) (actual time=0.176..0.239 rows=244 loops=3)
22	Filter: ((flag)::text = 'Greece'::text)
23	Rows Removed by Filter: 245
24	Planning Time: 0.293 ms
25	Execution Time: 1139.043 ms

```
6  EXPLAIN ANALYZE
7  SELECT V.id, V.type, count(V.type)
8  FROM vessels AS V
9  JOIN positions AS P ON V.id = P.vessel_id
10 WHERE P.speed > 30
11 GROUP BY V.id, V.type;
```

Data Output | Messages | Notifications

	QUERY PLAN text
1	Finalize GroupAggregate (cost=167497.51..167621.40 rows=489 width=75) (actual time=650.630..654.457 rows=18 loops=1)
2	Group Key: v.id
3	-> Gather Merge (cost=167497.51..167611.62 rows=978 width=75) (actual time=650.620..654.444 rows=44 loops=1)
4	Workers Planned: 2
5	Workers Launched: 2
6	-> Sort (cost=166497.49..166498.71 rows=489 width=75) (actual time=595.668..595.670 rows=15 loops=3)
7	Sort Key: v.id
8	Sort Method: quicksort Memory: 26kB
9	Worker 0: Sort Method: quicksort Memory: 26kB
10	Worker 1: Sort Method: quicksort Memory: 26kB
11	-> Partial HashAggregate (cost=166470.75..166475.64 rows=489 width=75) (actual time=595.613..595.618 rows=15 loops=3)
12	Group Key: v.id
13	Batches: 1 Memory Usage: 49kB
14	Worker 0: Batches: 1 Memory Usage: 49kB
15	Worker 1: Batches: 1 Memory Usage: 49kB
16	-> Hash Join (cost=18.00..166051.98 rows=83754 width=67) (actual time=2.478..581.597 rows=69914 loops=3)
17	Hash Cond: ((p.vessel_id)::text = (v.id)::text)
18	-> Parallel Seq Scan on positions p (cost=0.00..165812.22 rows=83754 width=65) (actual time=2.082..563.892 rows=69914 loop...
19	Filter: (speed > '30'::numeric)
20	Rows Removed by Filter: 2275637
21	-> Hash (cost=11.89..11.89 rows=489 width=67) (actual time=0.371..0.371 rows=489 loops=3)
22	Buckets: 1024 Batches: 1 Memory Usage: 56kB
23	-> Seq Scan on vessels v (cost=0.00..11.89 rows=489 width=67) (actual time=0.237..0.292 rows=489 loops=3)
24	Planning Time: 0.311 ms
25	Execution Time: 654.534 ms

```
6  EXPLAIN ANALYZE
7  SELECT P.t::date AS given_day, count(lon) AS registered_vessel_pisition
8  FROM positions AS P
9  JOIN vessels AS V on P.vessel_id = V.id
10 JOIN vessel_types AS VT ON  V.type = VT.code
11 WHERE VT.description LIKE 'Passenger,%'
12     AND P.t::date BETWEEN '2019-08-14' AND '2019-08-18'
13 GROUP BY P.t::date;
```

Data Output | Messages | Notifications

	QUERY PLAN text
1	Finalize GroupAggregate (cost=188962.40..189364.64 rows=3319 width=12) (actual time=619.505..632.376 rows=5 loops=1)
2	Group Key: ((p.t)::date)
3	-> Gather Merge (cost=188962.40..189309.32 rows=2766 width=12) (actual time=615.708..632.361 rows=15 loops=1)
4	Workers Planned: 2
5	Workers Launched: 2
6	-> Partial GroupAggregate (cost=187962.37..187990.03 rows=1383 width=12) (actual time=580.008..596.826 rows=5 loops=3)
7	Group Key: ((p.t)::date)
8	-> Sort (cost=187962.37..187965.83 rows=1383 width=12) (actual time=578.988..587.267 rows=120498 loops=3)
9	Sort Key: ((p.t)::date)
10	Sort Method: external merge Disk: 2672kB
11	Worker 0: Sort Method: external merge Disk: 2712kB
12	Worker 1: Sort Method: external merge Disk: 2688kB
13	-> Hash Join (cost=16.24..187890.22 rows=1383 width=12) (actual time=185.165..558.781 rows=120498 loops=3)
14	Hash Cond: ((p.vessel_id)::text = (v.id)::text)
15	-> Parallel Seq Scan on positions p (cost=0.00..187801.76 rows=14660 width=81) (actual time=184.371..500.422 rows=392404 loop...
16	Filter: (((t)::date >= '2019-08-14'::date) AND ((t)::date <= '2019-08-18'::date))
17	Rows Removed by Filter: 1953146
18	-> Hash (cost=15.67..15.67 rows=46 width=65) (actual time=0.725..0.727 rows=64 loops=3)
19	Buckets: 1024 Batches: 1 Memory Usage: 15kB
20	-> Hash Join (cost=2.45..15.67 rows=46 width=65) (actual time=0.633..0.714 rows=64 loops=3)
21	Hash Cond: ((v.type)::text = (vt.code)::text)
22	-> Seq Scan on vessels v (cost=0.00..11.89 rows=489 width=67) (actual time=0.368..0.405 rows=489 loops=3)
23	-> Hash (cost=2.33..2.33 rows=10 width=2) (actual time=0.251..0.252 rows=10 loops=3)
24	Buckets: 1024 Batches: 1 Memory Usage: 9kB
25	-> Seq Scan on vessel_types vt (cost=0.00..2.33 rows=10 width=2) (actual time=0.235..0.240 rows=10 loops=3)
26	Filter: ((description)::text ~~ 'Passenger,%'::text)
27	Rows Removed by Filter: 96
28	Planning Time: 0.405 ms
29	Execution Time: 633.667 ms

```
6  EXPLAIN ANALYZE
7  SELECT V.id as vessel_id, P.speed as stationary, V.type as cargo_vessels
8  FROM positions AS P
9  JOIN vessels AS V on P.vessel_id = V.id
10 WHERE V.type BETWEEN '70' and '79'
11     AND P.speed = 0.0
12     AND P.t >= '2019-08-15 00:00:00' AND P.t <= '2019-08-18 00:00:00'
13 GROUP BY V.id, P.speed
```

Data Output | Messages | Notifications

	QUERY PLAN text
1	Group (cost=182156.46..182181.49 rows=105 width=71) (actual time=534.606..538.364 rows=30 loops=1)
2	Group Key: v.id
3	-> Gather Merge (cost=182156.46..182180.96 rows=210 width=71) (actual time=534.605..538.349 rows=69 loops=1)
4	Workers Planned: 2
5	Workers Launched: 2
6	-> Sort (cost=181156.44..181156.70 rows=105 width=71) (actual time=481.087..481.089 rows=30 loops=3)
7	Sort Key: v.id
8	Sort Method: quicksort Memory: 27kB
9	Worker 0: Sort Method: quicksort Memory: 27kB
10	Worker 1: Sort Method: quicksort Memory: 27kB
11	-> Partial HashAggregate (cost=181151.86..181152.91 rows=105 width=71) (actual time=481.015..481.020 rows=30 loops=3)
12	Group Key: v.id
13	Batches: 1 Memory Usage: 24kB
14	Worker 0: Batches: 1 Memory Usage: 24kB
15	Worker 1: Batches: 1 Memory Usage: 24kB
16	-> Hash Join (cost=15.65..181039.89 rows=44791 width=71) (actual time=213.810..479.565 rows=7094 loops=3)
17	Hash Cond: ((p.vessel_id)::text = (v.id)::text)
18	-> Parallel Seq Scan on positions p (cost=0.00..180471.91 rows=208600 width=69) (actual time=183.338..463.558 rows=160205 loops=3)
19	Filter: ((t >= '2019-08-15 00:00:00'::timestamp without time zone) AND (t <= '2019-08-18 00:00:00'::timestamp without time zone) AND (s
20	Rows Removed by Filter: 2185346
21	-> Hash (cost=14.34..14.34 rows=105 width=67) (actual time=0.593..0.593 rows=104 loops=3)
22	Buckets: 1024 Batches: 1 Memory Usage: 16kB
23	-> Seq Scan on vessels v (cost=0.00..14.34 rows=105 width=67) (actual time=0.406..0.572 rows=104 loops=3)
24	Filter: (((type)::text >= '70'::text) AND ((type)::text <= '79'::text))
25	Rows Removed by Filter: 385
26	Planning Time: 0.313 ms
27	Execution Time: 538.402 ms

```
6  EXPLAIN ANALYZE
7  SELECT P.vessel_id as completely_stationary_vessels_between_12_and_18_August2019
8  FROM positions AS P
9  JOIN vessels AS V on P.vessel_id = V.id
10 WHERE V.type::integer BETWEEN 70 and 79
11     AND P.t >= '2019-08-12 00:00:00' AND P.t <= '2019-08-19 00:00:00'
12 GROUP BY P.vessel_id
13 HAVING MAX(P.speed) = 0.0;
```

Data Output | Messages | Notifications

	QUERY PLAN text
1	Finalize GroupAggregate (cost=175942.68..176072.80 rows=2 width=65) (actual time=536.140..538.259 rows=1 loops=1)
2	Group Key: p.vessel_id
3	Filter: (max(p.speed) = 0.0)
4	Rows Removed by Filter: 48
5	-> Gather Merge (cost=175942.68..176063.30 rows=844 width=97) (actual time=530.300..538.211 rows=143 loops=1)
6	Workers Planned: 2
7	Workers Launched: 2
8	-> Partial GroupAggregate (cost=174942.66..174965.86 rows=422 width=97) (actual time=504.871..512.060 rows=48 loops=3)
9	Group Key: p.vessel_id
10	-> Sort (cost=174942.66..174948.99 rows=2531 width=69) (actual time=504.804..506.773 rows=29614 loops=3)
11	Sort Key: p.vessel_id
12	Sort Method: quicksort Memory: 3612kB
13	Worker 0: Sort Method: quicksort Memory: 3525kB
14	Worker 1: Sort Method: quicksort Memory: 3498kB
15	-> Hash Join (cost=19.25..174799.59 rows=2531 width=69) (actual time=34.937..473.297 rows=29614 loops=3)
16	Hash Cond: ((p.vessel_id)::text = (v.id)::text)
17	-> Parallel Seq Scan on positions p (cost=0.00..173142.07 rows=618741 width=69) (actual time=29.804..426.931 rows=48868...
18	Filter: ((t >= '2019-08-12 00:00:00'::timestamp without time zone) AND (t <= '2019-08-19 00:00:00'::timestamp without time z...
19	Rows Removed by Filter: 1856870
20	-> Hash (cost=19.23..19.23 rows=2 width=65) (actual time=0.441..0.442 rows=104 loops=3)
21	Buckets: 1024 Batches: 1 Memory Usage: 16kB
22	-> Seq Scan on vessels v (cost=0.00..19.23 rows=2 width=65) (actual time=0.349..0.421 rows=104 loops=3)
23	Filter: (((type)::integer >= 70) AND ((type)::integer <= 79))
24	Rows Removed by Filter: 385
25	Planning Time: 0.294 ms
26	Execution Time: 538.717 ms

Παρατηρήσεις και συμπεράσματα

Το πράσινο χρώμα σημαίνει ότι ο χρόνος εκτέλεσης μειώθηκε από την προηγούμενη δοκιμή (στα αριστερά του), το κόκκινο χρώμα σημαίνει ότι ο χρόνος εκτέλεσης αυξήθηκε ενώ το πορτοκαλί σημαίνει ότι παρέμεινε ίδιος με πριν. Παρατηρούμε ότι κατά την αύξηση του μεγέθους των buffers, της PostgreSQL, ο χρόνος εκτέλεσης των queries μειώθηκε. Πιο συγκεκριμένα, έχουμε:

Query No. per shared_buffers size	128MB (default)	256MB	512MB	1024MB	2048MB
Query 1	890ms	924ms	943ms	894ms	850ms
Query 2	1222ms	1135ms	1163ms	1225ms	1139ms
Query 3	609ms	654ms	617ms	666ms	654ms
Query 4	664ms	624ms	619ms	720ms	633ms
Query 5A	522ms	508ms	547ms	510ms	538ms
Query 5B	600ms	523ms	532ms	528ms	538ms

Σύμφωνα με το documentation της PostgreSQL το χρήσιμο εύρος τιμών των shared_buffers είναι 65MB - 512MB ωστόσο εμείς στις μετρήσεις μας είδαμε μικρή βελτίωση και μετά από αυτό το εύρος τιμών.

Επίσης, σε περαιτέρω δοκιμές που έγιναν όπως βλέπουμε παρακάτω, παρατηρούμε ότι αν παραχωρήσουμε έως και την μισή μνήμη π.χ. 8GB τότε οι χρόνοι συνεχίζουν να μειώνονται. Για παράδειγμα παραχωρώντας 8GB RAM (αριστερό screenshot) και τρέχοντας το Query 5B βλέπουμε βελτίωση **52ms**. Παραχωρώντας το 50% την διαθέσιμη μνήμη (δεξί screenshot), στο ίδιο ερώτημα, η βελτίωση είναι **15ms** από τον χρόνο εκτέλεσης με shared_buffer = 128MB ενώ όταν παραχωρήσουμε το 100% της διαθέσιμης μνήμης **η βελτίωση είναι μηδενική ή παρατηρούμε καθυστερήσεις**. Συμπεραίνουμε ότι ξεπερνώντας το προτεινόμενο, από την PostgreSQL, 25% σε σχέση με την συνολική διαθέσιμη μνήμη του μηχανήματος, η μείωση των χρόνων εκτέλεσης παύει να αυξάνεται.

Shared_buffers = '8192MB';

shared_buffers = '16384MB';

```
EXPLAIN ANALYZE
SELECT P.vessel_id as completely_stationary_vessels_betweeen_12_and_18_August2019
FROM positions AS P
JOIN vessels AS V on P.vessel_id = V.id
WHERE V.type::integer BETWEEN 70 and 79
    AND P.t::date BETWEEN '2019-08-12' AND '2019-08-19'
GROUP BY P.vessel_id
HAVING MAX(P.speed) = 0;
```

Data Output | Messages | Notifications

	QUERY PLAN text
1	Finalize GroupAggregate (cost=189835.46..189926.52 rows=1 width=65) (actual time=1811.943..1855.262 rows=1 loops=1)
2	Group Key: p.vessel_id
3	Filter: (max(p.speed) = '0'::numeric)
4	Rows Removed by Filter: 52
5	-> Gather Merge (cost=189835.46..189924.12 rows=120 width=97) (actual time=1742.965..1855.152 rows=151 loops=1)
6	Workers Planned: 2
7	Workers Launched: 2
8	-> Partial GroupAggregate (cost=188835.44..188910.25 rows=60 width=97) (actual time=1608.833..1718.402 rows=50 loops=3)
9	Group Key: p.vessel_id
10	-> Merge Join (cost=188835.44..188909.35 rows=60 width=69) (actual time=1608.579..1713.807 rows=31533 loops=3)
11	Merge Cond: ((p.vessel_id)::text = (v.id)::text)
12	-> Sort (cost=188816.20..188852.85 rows=14660 width=69) (actual time=1607.836..1673.896 rows=523812 loops=3)
13	Sort Key: p.vessel_id
14	Sort Method: external merge Disk: 42104kB
15	Worker 0: Sort Method: external merge Disk: 42352kB
16	Worker 1: Sort Method: external merge Disk: 39408kB
17	-> Parallel Seq Scan on positions p (cost=0.00..187801.76 rows=14660 width=69) (actual time=0.016..524.695 rows=530768 loop...
18	Filter: (((t)::date >= '2019-08-12'::date) AND ((t)::date <= '2019-08-19'::date))
19	Rows Removed by Filter: 1814783
20	-> Sort (cost=19.24..19.24 rows=2 width=65) (actual time=0.737..0.751 rows=104 loops=3)
21	Sort Key: v.id
22	Sort Method: quicksort Memory: 33kB
23	Worker 0: Sort Method: quicksort Memory: 33kB
24	Worker 1: Sort Method: quicksort Memory: 33kB
25	-> Seq Scan on vessels v (cost=0.00..19.23 rows=2 width=65) (actual time=0.415..0.491 rows=104 loops=3)
26	Filter: (((type)::integer >= 70) AND ((type)::integer <= 79))
27	Rows Removed by Filter: 385
28	Planning Time: 0.370 ms
29	Execution Time: 1859.437 ms

Total rows: 29 of 29 | Query complete 00:00:01.878 | Ln 4, Co

shared_buffers ='32768MB';

```
EXPLAIN ANALYZE
SELECT P.vessel_id as completely_stationary_vessels_betweeen_12_and_18_August2019
FROM positions AS P
JOIN vessels AS V on P.vessel_id = V.id
WHERE V.type::integer BETWEEN 70 and 79
    AND P.t::date BETWEEN '2019-08-12' AND '2019-08-19'
GROUP BY P.vessel_id
HAVING MAX(P.speed) = 0;
```

Data Output | Messages | Notifications

	QUERY PLAN text
1	Finalize GroupAggregate (cost=189835.46..189926.52 rows=1 width=65) (actual time=1761.951..1802.183 rows=1 loops=1)
2	Group Key: p.vessel_id
3	Filter: (max(p.speed) = '0'::numeric)
4	Rows Removed by Filter: 52
5	-> Gather Merge (cost=189835.46..189924.12 rows=120 width=97) (actual time=1694.091..1802.082 rows=149 loops=1)
6	Workers Planned: 2
7	Workers Launched: 2
8	-> Partial GroupAggregate (cost=188835.44..188910.25 rows=60 width=97) (actual time=1559.314..1671.103 rows=50 loops=3)
9	Group Key: p.vessel_id
10	-> Merge Join (cost=188835.44..188909.35 rows=60 width=69) (actual time=1559.057..1666.511 rows=31583 loops=3)
11	Merge Cond: ((p.vessel_id)::text = (v.id)::text)
12	-> Sort (cost=188816.20..188852.85 rows=14660 width=69) (actual time=1558.465..1626.692 rows=523812 loops=3)
13	Sort Key: p.vessel_id
14	Sort Method: external merge Disk: 42232kB
15	Worker 0: Sort Method: external merge Disk: 42200kB
16	Worker 1: Sort Method: external merge Disk: 39400kB
17	-> Parallel Seq Scan on positions p (cost=0.00..187801.76 rows=14660 width=69) (actual time=0.011..480.066 rows=530768
18	Filter: (((t)::date >= '2019-08-12'::date) AND ((t)::date <= '2019-08-19'::date))
19	Rows Removed by Filter: 1814783
20	-> Sort (cost=19.24..19.24 rows=2 width=65) (actual time=0.587..0.601 rows=104 loops=3)
21	Sort Key: v.id
22	Sort Method: quicksort Memory: 33kB
23	Worker 0: Sort Method: quicksort Memory: 33kB
24	Worker 1: Sort Method: quicksort Memory: 33kB
25	-> Seq Scan on vessels v (cost=0.00..19.23 rows=2 width=65) (actual time=0.276..0.344 rows=104 loops=3)
26	Filter: (((type)::integer >= 70) AND ((type)::integer <= 79))
27	Rows Removed by Filter: 385
28	Planning Time: 0.800 ms
29	Execution Time: 1806.417 ms

Total rows: 29 of 29 | Query complete 00:00:01.825 | Ln

```
EXPLAIN ANALYZE
SELECT P.vessel_id as completely_stationary_vessels_betweeen_12_and_18_August2019
FROM positions AS P
JOIN vessels AS V on P.vessel_id = V.id
WHERE V.type::integer BETWEEN 70 and 79
    AND P.t::date BETWEEN '2019-08-12' AND '2019-08-19'
GROUP BY P.vessel_id
HAVING MAX(P.speed) = 0;
```

Data Output | Messages | Notifications

	QUERY PLAN text
1	Finalize GroupAggregate (cost=189835.46..189926.52 rows=1 width=65) (actual time=1783.926..1825.057 rows=1 loops=1)
2	Group Key: p.vessel_id
3	Filter: (max(p.speed) = '0'::numeric)
4	Rows Removed by Filter: 52
5	-> Gather Merge (cost=189835.46..189924.12 rows=120 width=97) (actual time=1709.631..1824.931 rows=147 loops=1)
6	Workers Planned: 2
7	Workers Launched: 2
8	-> Partial GroupAggregate (cost=188835.44..188910.25 rows=60 width=97) (actual time=1548.845..1658.596 rows=49 loops=3)
9	Group Key: p.vessel_id
10	-> Merge Join (cost=188835.44..188909.35 rows=60 width=69) (actual time=1548.587..1653.963 rows=31533 loops=3)
11	Merge Cond: ((p.vessel_id)::text = (v.id)::text)
12	-> Sort (cost=188816.20..188852.85 rows=14660 width=69) (actual time=1547.915..1614.256 rows=523812 loops=3)
13	Sort Key: p.vessel_id
14	Sort Method: external merge Disk: 42392kB
15	Worker 0: Sort Method: external merge Disk: 42032kB
16	Worker 1: Sort Method: external merge Disk: 39416kB
17	-> Parallel Seq Scan on positions p (cost=0.00..187801.76 rows=14660 width=69) (actual time=0.014..494.744 rows=530768 loop...
18	Filter: (((t)::date >= '2019-08-12'::date) AND ((t)::date <= '2019-08-19'::date))
19	Rows Removed by Filter: 1814783
20	-> Sort (cost=19.24..19.24 rows=2 width=65) (actual time=0.667..0.681 rows=104 loops=3)
21	Sort Key: v.id
22	Sort Method: quicksort Memory: 33kB
23	Worker 0: Sort Method: quicksort Memory: 33kB
24	Worker 1: Sort Method: quicksort Memory: 33kB
25	-> Seq Scan on vessels v (cost=0.00..19.23 rows=2 width=65) (actual time=0.350..0.425 rows=104 loops=3)
26	Filter: (((type)::integer >= 70) AND ((type)::integer <= 79))
27	Rows Removed by Filter: 385
28	Planning Time: 0.822 ms
29	Execution Time: 1830.299 ms

Total rows: 29 of 29 | Query complete 00:00:01.862 | Ln 4, C

Τέλος, παραχωρώντας πάνω από 32GB, δηλαδή περισσότερη μνήμη από την μνήμη RAM του υπολογιστή, για παράδειγμα 64GB μνήμης, η εντολή SHOW shared_buffers; σταμάτησε να λειτουργεί, το pgAdmin μετά από επανεκκίνηση σταμάτησε να συνδέεται στην PostgreSQL 16. Για την επαναφορά του στην αρχική τιμή χρειάστηκε να αλλάξουμε την τιμή χειροκίνητα στο αρχείο *postgresql.conf* όπως αναφέρουν οι οδηγίες. Όλες οι παραπάνω μετρήσεις έγιναν σε σύστημα Windows 11 με 32GB RAM και αφού πρώτα είχε εκτελεστεί η εντολή `VACUUM FULL` όπως ορίζουν οι οδηγίες της εργασίας.

Ερώτημα 3ο

Για την χρήση μεγαλύτερης επεξεργαστικής ισχύς από την default, χρησιμοποιούμε την εντολή *SHOW max_parallel_workers_per_gather;* Για να βρούμε την τρέχουσα τιμή των παράλληλων διεργασιών ενώ η εντολή *EXPLAIN* μας επιβεβαιώνει την παραπάνω τιμή κατά την επεξηγηματική εκτέλεση ενός query. Επίσης, η εντολή *SET max_parallel_workers_per_gather = <my_value>;* Χρησιμοποιείται για να αλλάξουμε αυτή την τιμή στην επιθυμητή. Για την κατανόηση της χρήσης των παραπάνω εντολών χρησιμοποιήθηκε το official documentation στο κεφάλαιο 20.4 Resource Consumption.

Για *max_parallel_workers_per_gather = 3* έχουμε τα εξής αποτελέσματα:

```
14 EXPLAIN ANALYZE
15 SELECT t::date as time_stamp, count(lon) as longtitude, count(lat) as latitude
16 FROM positions
17 GROUP BY time_stamp
18 ORDER BY longtitude desc;
19
```

Data Output | Messages | Notifications

	QUERY PLAN text
1	Sort (cost=640901.50..642037.53 rows=454411 width=20) (actual time=704.623..708.510 rows=24 loops=1)
2	Sort Key: (count(lon)) DESC
3	Sort Method: quicksort Memory: 26kB
4	-> Finalize GroupAggregate (cost=411660.53..588880.82 rows=454411 width=20) (actual time=704.597..708.504 rows=24 loops=1)
5	Group Key: ((t)::date)
6	-> Gather Merge (cost=411660.53..572976.44 rows=1363233 width=20) (actual time=704.593..708.489 rows=95 loops=1)
7	Workers Planned: 3
8	Workers Launched: 3
9	-> Sort (cost=410660.49..411796.52 rows=454411 width=20) (actual time=679.995..679.997 rows=24 loops=4)
10	Sort Key: ((t)::date)
11	Sort Method: quicksort Memory: 26kB
12	Worker 0: Sort Method: quicksort Memory: 26kB
13	Worker 1: Sort Method: quicksort Memory: 26kB
14	Worker 2: Sort Method: quicksort Memory: 26kB
15	-> Partial HashAggregate (cost=326359.44..358639.81 rows=454411 width=20) (actual time=679.819..679.976 rows=24 loops=4)
16	Group Key: (t)::date
17	Planned Partitions: 8 Batches: 1 Memory Usage: 1561kB
18	Worker 0: Batches: 1 Memory Usage: 1561kB
19	Worker 1: Batches: 1 Memory Usage: 1561kB
20	Worker 2: Batches: 1 Memory Usage: 1561kB
21	-> Parallel Seq Scan on positions (cost=0.00..157536.59 rows=2269887 width=20) (actual time=0.202..458.542 rows=1759163 loop...
22	Planning Time: 0.090 ms
23	Execution Time: 708.834 ms

Total rows: 23 of 23 Query complete 00:00:00.730

```
9  EXPLAIN ANALYZE
10 SELECT type as vessel_type, count(V.id) as vessels_with_greek_flag
11 FROM vessels AS V
12 JOIN positions AS P ON V.id = P.vessel_id
13 WHERE V.flag = 'Greece'
14 GROUP BY type;
```

Data Output | Messages | Notifications

	QUERY PLAN text
1	Finalize GroupAggregate (cost=164552.45..164564.99 rows=33 width=10) (actual time=907.588..911.610 rows=29 loops=1)
2	Group Key: v.type
3	-> Gather Merge (cost=164552.45..164564.16 rows=99 width=10) (actual time=907.582..911.588 rows=116 loops=1)
4	Workers Planned: 3
5	Workers Launched: 3
6	-> Sort (cost=163552.41..163552.49 rows=33 width=10) (actual time=882.882..882.884 rows=29 loops=4)
7	Sort Key: v.type
8	Sort Method: quicksort Memory: 26kB
9	Worker 0: Sort Method: quicksort Memory: 26kB
10	Worker 1: Sort Method: quicksort Memory: 26kB
11	Worker 2: Sort Method: quicksort Memory: 26kB
12	-> Partial HashAggregate (cost=163551.25..163551.58 rows=33 width=10) (actual time=882.821..882.825 rows=29 loops=4)
13	Group Key: v.type
14	Batches: 1 Memory Usage: 24kB
15	Worker 0: Batches: 1 Memory Usage: 24kB
16	Worker 1: Batches: 1 Memory Usage: 24kB
17	Worker 2: Batches: 1 Memory Usage: 24kB
18	-> Hash Join (cost=16.16..157888.13 rows=1132623 width=67) (actual time=0.440..668.723 rows=1368725 loops=4)
19	Hash Cond: ((p.vessel_id)::text = (v.id)::text)
20	-> Parallel Seq Scan on positions p (cost=0.00..151861.87 rows=2269887 width=65) (actual time=0.170..304.521 rows=1759163 loop...
21	-> Hash (cost=13.11..13.11 rows=244 width=67) (actual time=0.253..0.253 rows=244 loops=4)
22	Buckets: 1024 Batches: 1 Memory Usage: 32kB
23	-> Seq Scan on vessels v (cost=0.00..13.11 rows=244 width=67) (actual time=0.153..0.213 rows=244 loops=4)
24	Filter: ((flag)::text = 'Greece'::text)
25	Rows Removed by Filter: 245
26	Planning Time: 0.281 ms
27	Execution Time: 911.650 ms

Total rows: 27 of 27 Query complete 00:00:00.939

```
10 EXPLAIN ANALYZE
11 SELECT type, count(V.type)
12 FROM vessels AS V
13 JOIN positions AS P ON V.id = P.vessel_id
14 WHERE P.speed > 30
15 GROUP BY V.type;
```

Data Output | Messages | Notifications

	QUERY PLAN text
1	Finalize GroupAggregate (cost=159060.84..159073.38 rows=33 width=10) (actual time=520.467..524.812 rows=10 loops=1)
2	Group Key: v.type
3	-> Gather Merge (cost=159060.84..159072.55 rows=99 width=10) (actual time=520.460..524.803 rows=29 loops=1)
4	Workers Planned: 3
5	Workers Launched: 3
6	-> Sort (cost=158060.80..158060.88 rows=33 width=10) (actual time=491.638..491.639 rows=7 loops=4)
7	Sort Key: v.type
8	Sort Method: quicksort Memory: 25kB
9	Worker 0: Sort Method: quicksort Memory: 25kB
10	Worker 1: Sort Method: quicksort Memory: 25kB
11	Worker 2: Sort Method: quicksort Memory: 25kB
12	-> Partial HashAggregate (cost=158059.64..158059.97 rows=33 width=10) (actual time=491.603..491.605 rows=7 loops=4)
13	Group Key: v.type
14	Batches: 1 Memory Usage: 24kB
15	Worker 0: Batches: 1 Memory Usage: 24kB
16	Worker 1: Batches: 1 Memory Usage: 24kB
17	Worker 2: Batches: 1 Memory Usage: 24kB
18	-> Hash Join (cost=18.00..157729.45 rows=66038 width=2) (actual time=1.748..482.705 rows=52435 loops=4)
19	Hash Cond: ((p.vessel_id)::text = (v.id)::text)
20	-> Parallel Seq Scan on positions p (cost=0.00..157536.59 rows=66038 width=65) (actual time=1.365..470.313 rows=52435 loop...
21	Filter: (speed > '30'::numeric)
22	Rows Removed by Filter: 1706728
23	-> Hash (cost=11.89..11.89 rows=489 width=67) (actual time=0.360..0.361 rows=489 loops=4)
24	Buckets: 1024 Batches: 1 Memory Usage: 56kB
25	-> Seq Scan on vessels v (cost=0.00..11.89 rows=489 width=67) (actual time=0.211..0.270 rows=489 loops=4)
26	Planning Time: 0.558 ms
27	Execution Time: 524.896 ms

Total rows: 27 of 27 Query complete 00:00:00.547

```
9  EXPLAIN ANALYZE
10 SELECT P.t::date AS given_day, count(lon) AS registered_vessel_pisition
11 FROM positions AS P
12 JOIN vessels AS V on P.vessel_id = V.id
13 JOIN vessel_types AS VT ON  V.type = VT.code
14 WHERE VT.description LIKE 'Passenger,%'
15     AND P.t::date BETWEEN '2019-08-14' AND '2019-08-18'
16 GROUP BY P.t::date;
```

Data Output | Messages | Notifications

	QUERY PLAN text
1	GroupAggregate (cost=175686.84..176187.59 rows=3319 width=12) (actual time=522.819..595.969 rows=5 loops=1)
2	Group Key: ((p.t)::date)
3	-> Gather Merge (cost=175686.84..176079.50 rows=3319 width=12) (actual time=518.663..570.482 rows=361493 loops=1)
4	Workers Planned: 3
5	Workers Launched: 3
6	-> Sort (cost=174686.80..174689.48 rows=1071 width=12) (actual time=493.283..499.689 rows=90373 loops=4)
7	Sort Key: ((p.t)::date)
8	Sort Method: external merge Disk: 2040kB
9	Worker 0: Sort Method: external merge Disk: 2024kB
10	Worker 1: Sort Method: external merge Disk: 2008kB
11	Worker 2: Sort Method: external merge Disk: 2008kB
12	-> Hash Join (cost=16.24..174632.91 rows=1071 width=12) (actual time=36.195..476.936 rows=90373 loops=4)
13	Hash Cond: ((p.vessel_id)::text = (v.id)::text)
14	-> Parallel Seq Scan on positions p (cost=0.00..174560.75 rows=11349 width=81) (actual time=35.227..434.194 rows=29430
15	Filter: (((t)::date >= '2019-08-14'::date) AND ((t)::date <= '2019-08-18'::date))
16	Rows Removed by Filter: 1464860
17	-> Hash (cost=15.67..15.67 rows=46 width=65) (actual time=0.655..0.657 rows=64 loops=4)
18	Buckets: 1024 Batches: 1 Memory Usage: 15kB
19	-> Hash Join (cost=2.45..15.67 rows=46 width=65) (actual time=0.568..0.644 rows=64 loops=4)
20	Hash Cond: ((v.type)::text = (vt.code)::text)
21	-> Seq Scan on vessels v (cost=0.00..11.89 rows=489 width=67) (actual time=0.370..0.404 rows=489 loops=4)
22	-> Hash (cost=2.33..2.33 rows=10 width=2) (actual time=0.187..0.187 rows=10 loops=4)
23	Buckets: 1024 Batches: 1 Memory Usage: 9kB
24	-> Seq Scan on vessel_types vt (cost=0.00..2.33 rows=10 width=2) (actual time=0.174..0.179 rows=10 loops=4)
25	Filter: ((description)::text ~~ 'Passenger,%'::text)
26	Rows Removed by Filter: 96
27	Planning Time: 0.428 ms
28	Execution Time: 596.891 ms

Total rows: 28 of 28 Query complete 00:00:00.629

```
EXPLAIN ANALYZE
SELECT V.id as vessel_id, P.speed as stationary, V.type as cargo_vessels
FROM positions AS P
JOIN vessels AS V on P.vessel_id = V.id
WHERE V.type::integer BETWEEN 70 and 79
    AND P.speed = 0
    AND P.t BETWEEN '2019-08-15' AND '2019-08-18'
GROUP BY V.id, P.speed
```

Data Output | Messages | Notifications

	QUERY PLAN text
1	Group (cost=170360.09..170364.09 rows=2 width=71) (actual time=466.660..469.448 rows=90 loops=1)
2	Group Key: v.id
3	-> Gather Merge (cost=170360.09..170364.07 rows=6 width=71) (actual time=466.659..469.434 rows=117 loops=1)
4	Workers Planned: 3
5	Workers Launched: 3
6	-> Group (cost=169360.05..169363.33 rows=2 width=71) (actual time=442.542..443.118 rows=29 loops=4)
7	Group Key: v.id
8	-> Sort (cost=169360.05..169361.69 rows=655 width=71) (actual time=442.540..442.701 rows=5321 loops=4)
9	Sort Key: v.id
10	Sort Method: quicksort Memory: 707kB
11	Worker 0: Sort Method: quicksort Memory: 671kB
12	Worker 1: Sort Method: quicksort Memory: 717kB
13	Worker 2: Sort Method: quicksort Memory: 670kB
14	-> Hash Join (cost=19.25..169329.41 rows=655 width=71) (actual time=76.686..438.436 rows=5321 loops=4)
15	Hash Cond: ((p.vessel_id)::text = (v.id)::text)
16	-> Parallel Seq Scan on positions p (cost=0.00..168886.03 rows=160185 width=69) (actual time=74.098..426.032 rows=120154 loops=4)
17	Filter: ((t >= '2019-08-15 00:00:00'::timestamp without time zone) AND (t <= '2019-08-18 00:00:00'::timestamp without time zone) AND (speed =
18	Rows Removed by Filter: 1639009
19	-> Hash (cost=19.23..19.23 rows=2 width=67) (actual time=0.596..0.596 rows=104 loops=4)
20	Buckets: 1024 Batches: 1 Memory Usage: 19kB
21	-> Seq Scan on vessels v (cost=0.00..19.23 rows=2 width=67) (actual time=0.504..0.574 rows=104 loops=4)
22	Filter: (((type)::integer >= 70) AND ((type)::integer <= 79))
23	Rows Removed by Filter: 385
24	Planning Time: 0.304 ms
25	Execution Time: 469.558 ms

Total rows: 25 of 25 | Query complete 00:00:00.491 | Ln 11, Col

```
EXPLAIN ANALYZE
SELECT P.vessel_id as completely_stationary_vessels_betweeen_12_and_18_August2019
FROM positions AS P
JOIN vessels AS V on P.vessel_id = V.id
WHERE V.type::integer BETWEEN 70 and 79
    AND P.t::date BETWEEN '2019-08-12' AND '2019-08-19'
GROUP BY P.vessel_id
HAVING MAX(P.speed) = 0;
```

Data Output | Messages | Notifications

	QUERY PLAN text
1	Finalize GroupAggregate (cost=176344.39..176421.00 rows=1 width=65) (actual time=1378.069..1417.855 rows=1 loops=1)
2	Group Key: p.vessel_id
3	Filter: (max(p.speed) = '0'::numeric)
4	Rows Removed by Filter: 52
5	-> Gather Merge (cost=176344.39..176418.51 rows=138 width=97) (actual time=1319.681..1417.733 rows=198 loops=1)
6	Workers Planned: 3
7	Workers Launched: 3
8	-> Partial GroupAggregate (cost=175344.35..175402.26 rows=46 width=97) (actual time=1180.932..1281.019 rows=50 loops=4)
9	Group Key: p.vessel_id
10	-> Merge Join (cost=175344.35..175401.57 rows=46 width=69) (actual time=1180.594..1277.525 rows=23650 loops=4)
11	Merge Cond: ((p.vessel_id)::text = (v.id)::text)
12	-> Sort (cost=175325.12..175353.49 rows=11349 width=69) (actual time=1179.873..1246.439 rows=392859 loops=4)
13	Sort Key: p.vessel_id
14	Sort Method: external merge Disk: 30272kB
15	Worker 0: Sort Method: external merge Disk: 30224kB
16	Worker 1: Sort Method: external merge Disk: 32688kB
17	Worker 2: Sort Method: external merge Disk: 30712kB
18	-> Parallel Seq Scan on positions p (cost=0.00..174560.75 rows=11349 width=69) (actual time=61.125..441.656 rows=398076 loop...
19	Filter: (((t)::date >= '2019-08-12'::date) AND ((t)::date <= '2019-08-19'::date))
20	Rows Removed by Filter: 1361087
21	-> Sort (cost=19.24..19.24 rows=2 width=65) (actual time=0.716..0.728 rows=104 loops=4)
22	Sort Key: v.id
23	Sort Method: quicksort Memory: 33kB
24	Worker 0: Sort Method: quicksort Memory: 33kB
25	Worker 1: Sort Method: quicksort Memory: 33kB
26	Worker 2: Sort Method: quicksort Memory: 33kB
27	-> Seq Scan on vessels v (cost=0.00..19.23 rows=2 width=65) (actual time=0.388..0.467 rows=104 loops=4)
28	Filter: (((type)::integer >= 70) AND ((type)::integer <= 79))
29	Rows Removed by Filter: 385
30	Planning Time: 0.306 ms
31	Execution Time: 1421.288 ms

Total rows: 31 of 31 | Query complete 00:00:01.446

Για *max_parallel_workers_per_gather = 4* έχουμε τα εξής αποτελέσματα:

```
1  SHOW max_parallel_workers_per_gather;
```

Data Output | Messages | Notifications

	max_parallel_workers_per_gather text
1	4

```
12 EXPLAIN ANALYZE
13 SELECT t::date as time_stamp, count(lon) as longtitude, count(lat) as latitude
14 FROM positions
15 GROUP BY time_stamp
16 ORDER BY longtitude desc;
17
```

Data Output | Messages | Notifications

	QUERY PLAN text
1	Sort (cost=650274.45..651410.48 rows=454411 width=20) (actual time=665.020..669.370 rows=24 loops=1)
2	Sort Key: (count(lon)) DESC
3	Sort Method: quicksort Memory: 26kB
4	-> Finalize GroupAggregate (cost=361306.35..598253.77 rows=454411 width=20) (actual time=664.985..669.360 rows=24 loops=1)
5	Group Key: ((t)::date)
6	-> Gather Merge (cost=361306.35..578941.30 rows=1817644 width=20) (actual time=664.978..669.341 rows=118 loops=1)
7	Workers Planned: 4
8	Workers Launched: 4
9	-> Sort (cost=360306.29..361442.32 rows=454411 width=20) (actual time=617.187..617.189 rows=24 loops=5)
10	Sort Key: ((t)::date)
11	Sort Method: quicksort Memory: 26kB
12	Worker 0: Sort Method: quicksort Memory: 26kB
13	Worker 1: Sort Method: quicksort Memory: 26kB
14	Worker 2: Sort Method: quicksort Memory: 26kB
15	Worker 3: Sort Method: quicksort Memory: 26kB
16	-> Partial HashAggregate (cost=281990.28..308285.61 rows=454411 width=20) (actual time=616.898..617.164 rows=24 loops=5)
17	Group Key: (t)::date
18	Planned Partitions: 8 Batches: 1 Memory Usage: 1561kB
19	Worker 0: Batches: 1 Memory Usage: 1561kB
20	Worker 1: Batches: 1 Memory Usage: 1561kB
21	Worker 2: Batches: 1 Memory Usage: 1561kB
22	Worker 3: Batches: 1 Memory Usage: 1561kB
23	-> Parallel Seq Scan on positions (cost=0.00..151152.54 rows=1759163 width=20) (actual time=0.281..430.801 rows=1407330 loop...
24	Planning Time: 0.079 ms
25	Execution Time: 669.685 ms

Total rows: 25 of 25 | Query complete 00:00:00.702

```
EXPLAIN ANALYZE
SELECT type as vessel_type, count(V.id) as vessels_with_greek_flag
FROM vessels AS V
JOIN positions AS P ON V.id = P.vessel_id
WHERE V.flag = 'Greece'
GROUP BY type;
```

Data Output | Messages | Notifications

	QUERY PLAN text
1	Finalize GroupAggregate (cost=156818.74..156835.54 rows=33 width=10) (actual time=814.504..837.051 rows=29 loops=1)
2	Group Key: v.type
3	-> Gather Merge (cost=156818.74..156834.55 rows=132 width=10) (actual time=814.497..837.014 rows=145 loops=1)
4	Workers Planned: 4
5	Workers Launched: 4
6	-> Sort (cost=155818.69..155818.77 rows=33 width=10) (actual time=786.836..786.839 rows=29 loops=5)
7	Sort Key: v.type
8	Sort Method: quicksort Memory: 26kB
9	Worker 0: Sort Method: quicksort Memory: 26kB
10	Worker 1: Sort Method: quicksort Memory: 26kB
11	Worker 2: Sort Method: quicksort Memory: 26kB
12	Worker 3: Sort Method: quicksort Memory: 26kB
13	-> Partial HashAggregate (cost=155817.52..155817.85 rows=33 width=10) (actual time=786.764..786.769 rows=29 loops=5)
14	Group Key: v.type
15	Batches: 1 Memory Usage: 24kB
16	Worker 0: Batches: 1 Memory Usage: 24kB
17	Worker 1: Batches: 1 Memory Usage: 24kB
18	Worker 2: Batches: 1 Memory Usage: 24kB
19	Worker 3: Batches: 1 Memory Usage: 24kB
20	-> Hash Join (cost=16.16..151428.61 rows=877783 width=67) (actual time=0.603..608.538 rows=1094980 loops=5)
21	Hash Cond: ((p.vessel_id)::text = (v.id)::text)
22	-> Parallel Seq Scan on positions p (cost=0.00..146754.63 rows=1759163 width=65) (actual time=0.259..306.772 rows=1407330 loop.
23	-> Hash (cost=13.11..13.11 rows=244 width=67) (actual time=0.325..0.325 rows=244 loops=5)
24	Buckets: 1024 Batches: 1 Memory Usage: 32kB
25	-> Seq Scan on vessels v (cost=0.00..13.11 rows=244 width=67) (actual time=0.214..0.279 rows=244 loops=5)
26	Filter: ((flag)::text = 'Greece'::text)
27	Rows Removed by Filter: 245
28	Planning Time: 0.285 ms
29	Execution Time: 837.091 ms

Total rows: 29 of 29 Query complete 00:00:00.870

```
EXPLAIN ANALYZE
SELECT type, count(V.type)
FROM vessels AS V
JOIN positions AS P ON V.id = P.vessel_id
WHERE P.speed > 30
GROUP BY V.type;
```

Data Output | Messages | Notifications

	QUERY PLAN text
1	Finalize GroupAggregate (cost=152563.17..152579.97 rows=33 width=10) (actual time=500.362..504.290 rows=10 loops=1)
2	Group Key: v.type
3	-> Gather Merge (cost=152563.17..152578.98 rows=132 width=10) (actual time=500.355..504.279 rows=34 loops=1)
4	Workers Planned: 4
5	Workers Launched: 4
6	-> Sort (cost=151563.11..151563.20 rows=33 width=10) (actual time=471.752..471.754 rows=7 loops=5)
7	Sort Key: v.type
8	Sort Method: quicksort Memory: 25kB
9	Worker 0: Sort Method: quicksort Memory: 25kB
10	Worker 1: Sort Method: quicksort Memory: 25kB
11	Worker 2: Sort Method: quicksort Memory: 25kB
12	Worker 3: Sort Method: quicksort Memory: 25kB
13	-> Partial HashAggregate (cost=151561.95..151562.28 rows=33 width=10) (actual time=471.721..471.723 rows=7 loops=5)
14	Group Key: v.type
15	Batches: 1 Memory Usage: 24kB
16	Worker 0: Batches: 1 Memory Usage: 24kB
17	Worker 1: Batches: 1 Memory Usage: 24kB
18	Worker 2: Batches: 1 Memory Usage: 24kB
19	Worker 3: Batches: 1 Memory Usage: 24kB
20	-> Hash Join (cost=18.60..151306.05 rows=51180 width=2) (actual time=2.411..463.942 rows=41948 loops=5)
21	Hash Cond: ((p.vessel_id)::text = (v.id)::text)
22	-> Parallel Seq Scan on positions p (cost=0.00..151152.53 rows=51180 width=65) (actual time=1.951..452.977 rows=41948 loop.
23	Filter: (speed > '30'::numeric)
24	Rows Removed by Filter: 1365382
25	-> Hash (cost=11.89..11.89 rows=489 width=67) (actual time=0.436..0.437 rows=489 loops=5)
26	Buckets: 1024 Batches: 1 Memory Usage: 56kB
27	-> Seq Scan on vessels v (cost=0.00..11.89 rows=489 width=67) (actual time=0.286..0.345 rows=489 loops=5)
28	Planning Time: 0.315 ms
29	Execution Time: 504.351 ms

Total rows: 29 of 29 Query complete 00:00:00.541

```
EXPLAIN ANALYZE
SELECT P.t::date AS given_day, count(lon) AS registered_vessel_pisition
FROM positions AS P
JOIN vessels AS V on P.vessel_id = V.id
JOIN vessel_types AS VT ON  V.type = VT.code
WHERE VT.description LIKE 'Passenger,%'
    AND P.t::date BETWEEN '2019-08-14' AND '2019-08-18'
GROUP BY P.t::date;
```

Data Output | Messages | Notifications

	QUERY PLAN text
1	GroupAggregate (cost=165446.13..165901.61 rows=3319 width=12) (actual time=497.657..568.208 rows=5 loops=1)
2	Group Key: ((p.t)::date)
3	-> Gather Merge (cost=165446.13..165843.53 rows=3319 width=12) (actual time=493.718..541.411 rows=361493 loops=1)
4	Workers Planned: 4
5	Workers Launched: 4
6	-> Sort (cost=164446.07..164448.15 rows=830 width=12) (actual time=463.291..468.889 rows=72299 loops=5)
7	Sort Key: ((p.t)::date)
8	Sort Method: external merge Disk: 1576kB
9	Worker 0: Sort Method: external merge Disk: 1640kB
10	Worker 1: Sort Method: external merge Disk: 1672kB
11	Worker 2: Sort Method: external merge Disk: 1576kB
12	Worker 3: Sort Method: external merge Disk: 1632kB
13	-> Hash Join (cost=16.24..164405.83 rows=830 width=12) (actual time=103.277..449.796 rows=72299 loops=5)
14	Hash Cond: ((p.vessel_id)::text = (v.id)::text)
15	-> Parallel Seq Scan on positions p (cost=0.00..164346.26 rows=8796 width=81) (actual time=102.304..413.378 rows=235442 loop...
16	Filter: (((t)::date >= '2019-08-14'::date) AND ((t)::date <= '2019-08-18'::date))
17	Rows Removed by Filter: 1171888
18	-> Hash (cost=15.67..15.67 rows=46 width=65) (actual time=0.833..0.835 rows=64 loops=5)
19	Buckets: 1024 Batches: 1 Memory Usage: 15kB
20	-> Hash Join (cost=2.45..15.67 rows=46 width=65) (actual time=0.718..0.814 rows=64 loops=5)
21	Hash Cond: ((v.type)::text = (vt.code)::text)
22	-> Seq Scan on vessels v (cost=0.00..11.89 rows=489 width=67) (actual time=0.437..0.478 rows=489 loops=5)
23	-> Hash (cost=2.33..2.33 rows=10 width=2) (actual time=0.265..0.265 rows=10 loops=5)
24	Buckets: 1024 Batches: 1 Memory Usage: 9kB
25	-> Seq Scan on vessel_types vt (cost=0.00..2.33 rows=10 width=2) (actual time=0.248..0.253 rows=10 loops=5)
26	Filter: ((description)::text ~~ 'Passenger,%'::text)
27	Rows Removed by Filter: 96
28	Planning Time: 0.441 ms
29	Execution Time: 569.258 ms

Total rows: 29 of 29 | Query complete 00:00:00.600

```
EXPLAIN ANALYZE
SELECT V.id as vessel_id, P.speed as stationary, V.type as cargo_vessels
FROM positions AS P
JOIN vessels AS V on P.vessel_id = V.id
WHERE V.type::integer BETWEEN 70 and 79
    AND P.speed = 0
    AND P.t BETWEEN '2019-08-15' AND '2019-08-18'
GROUP BY V.id, P.speed
```

Data Output | Messages | Notifications

	QUERY PLAN text
1	Group (cost=161319.19..161322.70 rows=2 width=71) (actual time=447.760..453.142 rows=30 loops=1)
2	Group Key: v.id
3	-> Gather Merge (cost=161319.19..161322.68 rows=8 width=71) (actual time=447.758..453.120 rows=140 loops=1)
4	Workers Planned: 4
5	Workers Launched: 4
6	-> Group (cost=160319.13..160321.67 rows=2 width=71) (actual time=397.839..398.440 rows=28 loops=5)
7	Group Key: v.id
8	-> Sort (cost=160319.13..160320.40 rows=508 width=71) (actual time=397.835..398.031 rows=4257 loops=5)
9	Sort Key: v.id
10	Sort Method: quicksort Memory: 611kB
11	Worker 0: Sort Method: quicksort Memory: 631kB
12	Worker 1: Sort Method: quicksort Memory: 399kB
13	Worker 2: Sort Method: quicksort Memory: 608kB
14	Worker 3: Sort Method: quicksort Memory: 613kB
15	-> Hash Join (cost=19.25..160296.80 rows=508 width=71) (actual time=71.458..394.466 rows=4257 loops=5)
16	Hash Cond: ((p.vessel_id)::text = (v.id)::text)
17	-> Parallel Seq Scan on positions p (cost=0.00..159948.35 rows=124143 width=69) (actual time=69.311..384.357 rows=96123 loops=5)
18	Filter: ((t >= '2019-08-15 00:00:00'::timestamp without time zone) AND (t <= '2019-08-18 00:00:00'::timestamp without time zone) AND (speed = ...
19	Rows Removed by Filter: 1311207
20	-> Hash (cost=19.23..19.23 rows=2 width=67) (actual time=0.574..0.575 rows=104 loops=5)
21	Buckets: 1024 Batches: 1 Memory Usage: 19kB
22	-> Seq Scan on vessels v (cost=0.00..19.23 rows=2 width=67) (actual time=0.463..0.544 rows=104 loops=5)
23	Filter: (((type)::integer >= 70) AND ((type)::integer <= 79))
24	Rows Removed by Filter: 385
25	Planning Time: 0.308 ms
26	Execution Time: 453.253 ms

Total rows: 26 of 26 | Query complete 00:00:00.471 | Ln 10, Col

```
EXPLAIN ANALYZE
SELECT P.vessel_id as completely_stationary_vessels_betweeen_12_and_18_August2019
FROM positions AS P
JOIN vessels AS V on P.vessel_id = V.id
WHERE V.type::integer BETWEEN 70 and 79
    AND P.t::date BETWEEN '2019-08-12' AND '2019-08-19'
GROUP BY P.vessel_id
HAVING MAX(P.speed) = 0;
```

Data Output | Messages | Notifications

	QUERY PLAN text
1	GroupAggregate (cost=165941.80..166005.82 rows=1 width=65) (actual time=1196.291..1269.886 rows=1 loops=1)
2	Group Key: p.vessel_id
3	Filter: (max(p.speed) = '0'::numeric)
4	Rows Removed by Filter: 52
5	-> Gather Merge (cost=165941.80..166003.30 rows=144 width=69) (actual time=1060.472..1255.510 rows=94599 loops=1)
6	Workers Planned: 4
7	Workers Launched: 4
8	-> Merge Join (cost=164941.74..164986.09 rows=36 width=69) (actual time=1023.882..1104.860 rows=18920 loops=5)
9	Merge Cond: ((p.vessel_id)::text = (v.id)::text)
10	-> Sort (cost=164922.51..164944.50 rows=8796 width=69) (actual time=1023.182..1079.977 rows=314287 loops=5)
11	Sort Key: p.vessel_id
12	Sort Method: external merge Disk: 26528kB
13	Worker 0: Sort Method: external merge Disk: 23712kB
14	Worker 1: Sort Method: external merge Disk: 23672kB
15	Worker 2: Sort Method: external merge Disk: 23696kB
16	Worker 3: Sort Method: external merge Disk: 26312kB
17	-> Parallel Seq Scan on positions p (cost=0.00..164946.26 rows=8796 width=69) (actual time=0.261..407.016 rows=318461 loop...
18	Filter: (((t)::date >= '2019-08-12'::date) AND ((t)::date <= '2019-08-19'::date))
19	Rows Removed by Filter: 1088870
20	-> Sort (cost=19.24..19.24 rows=2 width=65) (actual time=0.694..0.709 rows=104 loops=5)
21	Sort Key: v.id
22	Sort Method: quicksort Memory: 33kB
23	Worker 0: Sort Method: quicksort Memory: 33kB
24	Worker 1: Sort Method: quicksort Memory: 33kB
25	Worker 2: Sort Method: quicksort Memory: 33kB
26	Worker 3: Sort Method: quicksort Memory: 33kB
27	-> Seq Scan on vessels v (cost=0.00..19.23 rows=2 width=65) (actual time=0.377..0.450 rows=104 loops=5)
28	Filter: (((type)::integer >= 70) AND ((type)::integer <= 79))
29	Rows Removed by Filter: 385
30	Planning Time: 0.322 ms
31	Execution Time: 1273.115 ms

Total rows: 31 of 31 | Query complete 00:00:01.304

Για *max_parallel_workers_per_gather = 5* έχουμε τα εξής αποτελέσματα:

```
SHOW max_parallel_workers_per_gather;
```

Data Output | Messages | Notifications

	max_parallel_workers_per_gather text
1	5

```
9  EXPLAIN ANALYZE
10 SELECT t::date as time_stamp, count(lon) as longtitude, count(lat) as latitude
11 FROM positions
12 GROUP BY time_stamp
13 ORDER BY longtitude desc;
```

Data Output | Messages | Notifications

	QUERY PLAN text
1	Sort (cost=676106.89..677242.92 rows=454411 width=20) (actual time=628.803..633.801 rows=24 loops=1)
2	Sort Key: (count(lon)) DESC
3	Sort Method: quicksort Memory: 26kB
4	-> Finalize GroupAggregate (cost=326617.84..624086.21 rows=454411 width=20) (actual time=628.252..633.292 rows=24 loops=1)
5	Group Key: ((t)::date)
6	-> Gather Merge (cost=326617.84..601365.66 rows=2272055 width=20) (actual time=628.245..638.264 rows=141 loops=1)
7	Workers Planned: 5
8	Workers Launched: 5
9	-> Sort (cost=325617.76..326753.79 rows=454411 width=20) (actual time=593.134..593.136 rows=24 loops=6)
10	Sort Key: ((t)::date)
11	Sort Method: quicksort Memory: 26kB
12	Worker 0: Sort Method: quicksort Memory: 26kB
13	Worker 1: Sort Method: quicksort Memory: 26kB
14	Worker 2: Sort Method: quicksort Memory: 26kB
15	Worker 3: Sort Method: quicksort Memory: 26kB
16	Worker 4: Sort Method: quicksort Memory: 26kB
17	-> Partial HashAggregate (cost=251424.80..273597.08 rows=454411 width=20) (actual time=592.800..593.111 rows=24 loops=6)
18	Group Key: (t)::date
19	Planned Partitions: 8 Batches: 1 Memory Usage: 1561kB
20	Worker 0: Batches: 1 Memory Usage: 1561kB
21	Worker 1: Batches: 1 Memory Usage: 1561kB
22	Worker 2: Batches: 1 Memory Usage: 1561kB
23	Worker 3: Batches: 1 Memory Usage: 1561kB
24	Worker 4: Batches: 1 Memory Usage: 1561kB
25	-> Parallel Seq Scan on positions (cost=0.00..146754.63 rows=1407330 width=20) (actual time=0.280..430.844 rows=1172775 loop...
26	Planning Time: 0.102 ms
27	Execution Time: 633.628 ms

Total rows: 27 of 27 Query complete 00:00:00.669

```
4  EXPLAIN ANALYZE
5  SELECT type as vessel_type, count(V.id) as vessels_with_greek_flag
6  FROM vessels AS V
7  JOIN positions AS P ON V.id = P.vessel_id
8  WHERE V.flag = 'Greece'
9  GROUP BY type;
```

Data Output | Messages | Notifications

	QUERY PLAN text
1	Finalize GroupAggregate (cost=151491.09..151512.20 rows=33 width=10) (actual time=766.694..771.564 rows=29 loops=1)
2	Group Key: v.type
3	-> Gather Merge (cost=151491.09..151511.05 rows=165 width=10) (actual time=766.685..771.588 rows=174 loops=1)
4	Workers Planned: 5
5	Workers Launched: 5
6	-> Sort (cost=150491.02..150491.10 rows=33 width=10) (actual time=692.748..692.750 rows=29 loops=6)
7	Sort Key: v.type
8	Sort Method: quicksort Memory: 26kB
9	Worker 0: Sort Method: quicksort Memory: 26kB
10	Worker 1: Sort Method: quicksort Memory: 26kB
11	Worker 2: Sort Method: quicksort Memory: 26kB
12	Worker 3: Sort Method: quicksort Memory: 26kB
13	Worker 4: Sort Method: quicksort Memory: 26kB
14	-> Partial HashAggregate (cost=150489.85..150490.18 rows=33 width=10) (actual time=692.676..692.680 rows=29 loops=6)
15	Group Key: v.type
16	Batches: 1 Memory Usage: 24kB
17	Worker 0: Batches: 1 Memory Usage: 24kB
18	Worker 1: Batches: 1 Memory Usage: 24kB
19	Worker 2: Batches: 1 Memory Usage: 24kB
20	Worker 3: Batches: 1 Memory Usage: 24kB
21	Worker 4: Batches: 1 Memory Usage: 24kB
22	-> Hash Join (cost=16.16..146978.72 rows=702226 width=67) (actual time=0.672..538.986 rows=912483 loops=6)
23	Hash Cond: ((p.vessel_id)::text = (v.id)::text)
24	-> Parallel Seq Scan on positions p (cost=0.00..143236.30 rows=1407330 width=65) (actual time=0.306..275.290 rows=1172775 loop...
25	-> Hash (cost=13.11..13.11 rows=244 width=67) (actual time=0.337..0.337 rows=244 loops=6)
26	Buckets: 1024 Batches: 1 Memory Usage: 32kB
27	-> Seq Scan on vessels v (cost=0.00..13.11 rows=244 width=67) (actual time=0.233..0.294 rows=244 loops=6)
28	Filter: ((flag)::text = 'Greece'::text)
29	Rows Removed by Filter: 245
30	Planning Time: 0.302 ms
31	Execution Time: 771.606 ms

✓ Successfully run. Total query runtime: 829 msec. 31 rows affe

Total rows: 31 of 31 Query complete 00:00:00.829

```
EXPLAIN ANALYZE
SELECT type, count(V.type)
FROM vessels AS V
JOIN positions AS P ON V.id = P.vessel_id
WHERE P.speed > 30
GROUP BY V.type;
```

Data Output | Messages | Notifications

	QUERY PLAN text
1	Finalize GroupAggregate (cost=148087.00..148108.11 rows=33 width=10) (actual time=508.100..512.650 rows=10 loops=1)
2	Group Key: v.type
3	-> Gather Merge (cost=148087.00..148106.95 rows=165 width=10) (actual time=508.093..512.640 rows=36 loops=1)
4	Workers Planned: 5
5	Workers Launched: 5
6	-> Sort (cost=147086.92..147087.01 rows=33 width=10) (actual time=474.527..474.529 rows=6 loops=6)
7	Sort Key: v.type
8	Sort Method: quicksort Memory: 25kB
9	Worker 0: Sort Method: quicksort Memory: 25kB
10	Worker 1: Sort Method: quicksort Memory: 25kB
11	Worker 2: Sort Method: quicksort Memory: 25kB
12	Worker 3: Sort Method: quicksort Memory: 25kB
13	Worker 4: Sort Method: quicksort Memory: 25kB
14	-> Partial HashAggregate (cost=147085.76..147086.09 rows=33 width=10) (actual time=474.493..474.495 rows=6 loops=6)
15	Group Key: v.type
16	Batches: 1 Memory Usage: 24kB
17	Worker 0: Batches: 1 Memory Usage: 24kB
18	Worker 1: Batches: 1 Memory Usage: 24kB
19	Worker 2: Batches: 1 Memory Usage: 24kB
20	Worker 3: Batches: 1 Memory Usage: 24kB
21	Worker 4: Batches: 1 Memory Usage: 24kB
22	-> Hash Join (cost=18.00..146881.04 rows=40944 width=2) (actual time=1.167..467.671 rows=34957 loops=6)
23	Hash Cond: ((p.vessel_id)::text = (v.id)::text)
24	-> Parallel Seq Scan on positions p (cost=0.00..146754.63 rows=40944 width=65) (actual time=0.757..458.002 rows=34957 loop...
25	Filter: (speed > '30'::numeric)
26	Rows Removed by Filter: 1137818
27	-> Hash (cost=11.89..11.89 rows=489 width=67) (actual time=0.382..0.382 rows=489 loops=6)
28	Buckets: 1024 Batches: 1 Memory Usage: 56kB
29	-> Seq Scan on vessels v (cost=0.00..11.89 rows=489 width=67) (actual time=0.242..0.297 rows=489 loops=6)
30	Planning Time: 0.287 ms
31	Execution Time: 512.685 ms

Total rows: 31 of 31 Query complete 00:00:00.546

```
EXPLAIN ANALYZE
SELECT P.t::date AS given_day, count(lon) AS registered_vessel_pisition
FROM positions AS P
JOIN vessels AS V on P.vessel_id = V.id
JOIN vessel_types AS VT ON  V.type = VT.code
WHERE VT.description LIKE 'Passenger,%'
    AND P.t::date BETWEEN '2019-08-14' AND '2019-08-18'
```

Data Output | Messages | Notifications

	QUERY PLAN text
1	GroupAggregate (cost=158391.72..158851.15 rows=3319 width=12) (actual time=489.211..563.088 rows=5 loops=1)
2	Group Key: ((p.t)::date)
3	-> Gather Merge (cost=158391.72..158793.07 rows=3319 width=12) (actual time=484.954..536.340 rows=361403 loops=1)
4	Workers Planned: 5
5	Workers Launched: 5
6	-> Sort (cost=157391.64..157393.30 rows=664 width=12) (actual time=450.715..452.922 rows=60249 loops=6)
7	Sort Key: ((p.t)::date)
8	Sort Method: quicksort Memory: 3956kB
9	Worker 0: Sort Method: quicksort Memory: 3895kB
10	Worker 1: Sort Method: quicksort Memory: 3676kB
11	Worker 2: Sort Method: quicksort Memory: 3957kB
12	Worker 3: Sort Method: quicksort Memory: 3751kB
13	Worker 4: Sort Method: quicksort Memory: 3905kB
14	-> Hash Join (cost=16.24..157360.52 rows=664 width=12) (actual time=51.122..443.157 rows=60249 loops=6)
15	Hash Cond: ((p.vessel_id)::text = (v.id)::text)
16	-> Parallel Seq Scan on positions p (cost=0.00..157309.60 rows=7037 width=81) (actual time=50.063..411.843 rows=196202 loop...
17	Filter: ((((t)::date >= '2019-08-14'::date) AND ((t)::date <= '2019-08-18'::date))
18	Rows Removed by Filter: 976573
19	-> Hash (cost=15.67..15.67 rows=46 width=65) (actual time=0.906..0.909 rows=64 loops=6)
20	Buckets: 1024 Batches: 1 Memory Usage: 15kB
21	-> Hash Join (cost=2.45..15.67 rows=46 width=65) (actual time=0.817..0.894 rows=64 loops=6)
22	Hash Cond: ((v.type)::text = (vt.code)::text)
23	-> Seq Scan on vessels v (cost=0.00..11.89 rows=489 width=67) (actual time=0.565..0.599 rows=489 loops=6)
24	-> Hash (cost=2.33..2.33 rows=10 width=2) (actual time=0.239..0.240 rows=10 loops=6)
25	Buckets: 1024 Batches: 1 Memory Usage: 9kB
26	-> Seq Scan on vessel_types vt (cost=0.00..2.33 rows=10 width=2) (actual time=0.224..0.229 rows=10 loops=6)
27	Filter: ((description)::text ~~ 'Passenger,%'::text)
28	Rows Removed by Filter: 96
29	Planning Time: 0.427 ms
30	Execution Time: 563.496 ms

Total rows: 30 of 30 Query complete 00:00:00.611

```
EXPLAIN ANALYZE
SELECT V.id as vessel_id, P.speed as stationary, V.type as cargo_vessels
FROM positions AS P
JOIN vessels AS V on P.vessel_id = V.id
WHERE V.type::integer BETWEEN 70 and 79
    AND P.speed = 0
    AND P.t BETWEEN '2019-08-15' AND '2019-08-18'
GROUP BY V.id, P.speed
```

Data Output | Messages | Notifications

	QUERY PLAN text
1	Group (cost=155091.15..155094.41 rows=2 width=71) (actual time=440.319..444.667 rows=30 loops=1)
2	Group Key: v.id
3	-> Gather Merge (cost=155091.15..155094.39 rows=10 width=71) (actual time=440.318..444.642 rows=173 loops=1)
4	Workers Planned: 5
5	Workers Launched: 5
6	-> Group (cost=154091.08..154093.11 rows=2 width=71) (actual time=409.528..409.927 rows=29 loops=6)
7	Group Key: v.id
8	-> Sort (cost=154091.08..154092.09 rows=406 width=71) (actual time=409.526..409.639 rows=3547 loops=6)
9	Sort Key: v.id
10	Sort Method: quicksort Memory: 450kB
11	Worker 0: Sort Method: quicksort Memory: 426kB
12	Worker 1: Sort Method: quicksort Memory: 448kB
13	Worker 2: Sort Method: quicksort Memory: 434kB
14	Worker 3: Sort Method: quicksort Memory: 397kB
15	Worker 4: Sort Method: quicksort Memory: 424kB
16	-> Hash Join (cost=19.25..154073.49 rows=406 width=71) (actual time=58.014..406.640 rows=3547 loops=6)
17	Hash Cond: ((p.vessel_id)::text = (v.id)::text)
18	-> Parallel Seq Scan on positions p (cost=0.00..153791.28 rows=99314 width=69) (actual time=55.474..397.557 rows=80102 loops=6)
19	Filter: ((t >= '2019-08-15 00:00:00'::timestamp without time zone) AND (t <= '2019-08-18 00:00:00'::timestamp without time zone) AND (speed =
20	Rows Removed by Filter: 1092673
21	-> Hash (cost=19.23..19.23 rows=2 width=67) (actual time=0.738..0.738 rows=104 loops=6)
22	Buckets: 1024 Batches: 1 Memory Usage: 19kB
23	-> Seq Scan on vessels v (cost=0.00..19.23 rows=2 width=67) (actual time=0.639..0.713 rows=104 loops=6)
24	Filter: (((type)::integer >= 70) AND ((type)::integer <= 79))
25	Rows Removed by Filter: 385
26	Planning Time: 0.307 ms
27	Execution Time: 444.772 ms

Total rows: 27 of 27 | Query complete 00:00:00.472 | Ln 5, Co

```
SELECT P.vessel_id as completely_stationary_vessels_betweeen_12_and_18_August2019
FROM positions AS P
JOIN vessels AS V on P.vessel_id = V.id
WHERE V.type::integer BETWEEN 70 and 79
    AND P.t::date BETWEEN '2019-08-12' AND '2019-08-19'
GROUP BY P.vessel_id
HAVING MAX(P.speed) = 0;
```

Data Output | Messages | Notifications

	QUERY PLAN text
1	GroupAggregate (cost=158778.61..158833.95 rows=1 width=65) (actual time=1105.371..1173.050 rows=1 loops=1)
2	Group Key: p.vessel_id
3	Filter: (max(p.speed) = '0'::numeric)
4	Rows Removed by Filter: 52
5	-> Gather Merge (cost=158778.61..158831.43 rows=144 width=69) (actual time=981.130..1158.845 rows=94599 loops=1)
6	Workers Planned: 5
7	Workers Launched: 5
8	-> Merge Join (cost=157778.53..157814.01 rows=29 width=69) (actual time=921.887..988.596 rows=15766 loops=6)
9	Merge Cond: ((p.vessel_id)::text = (v.id)::text)
10	-> Sort (cost=157759.29..157776.89 rows=7037 width=69) (actual time=921.079..967.935 rows=261906 loops=6)
11	Sort Key: p.vessel_id
12	Sort Method: external merge Disk: 21328kB
13	Worker 0: Sort Method: external merge Disk: 19400kB
14	Worker 1: Sort Method: external merge Disk: 22016kB
15	Worker 2: Sort Method: external merge Disk: 20952kB
16	Worker 3: Sort Method: external merge Disk: 19376kB
17	Worker 4: Sort Method: external merge Disk: 20840kB
18	-> Parallel Seq Scan on positions p (cost=0.00..157309.60 rows=7037 width=69) (actual time=49.585..381.898 rows=265364 loop...
19	Filter: (((t)::date >= '2019-08-12'::date) AND ((t)::date <= '2019-08-19'::date))
20	Rows Removed by Filter: 907391
21	-> Sort (cost=19.24..19.24 rows=2 width=65) (actual time=0.802..0.816 rows=104 loops=6)
22	Sort Key: v.id
23	Sort Method: quicksort Memory: 33kB
24	Worker 0: Sort Method: quicksort Memory: 33kB
25	Worker 1: Sort Method: quicksort Memory: 33kB
26	Worker 2: Sort Method: quicksort Memory: 33kB
27	Worker 3: Sort Method: quicksort Memory: 33kB
28	Worker 4: Sort Method: quicksort Memory: 33kB
29	-> Seq Scan on vessels v (cost=0.00..19.23 rows=2 width=65) (actual time=0.464..0.541 rows=104 loops=6)
30	Filter: (((type)::integer >= 70) AND ((type)::integer <= 79))
31	Rows Removed by Filter: 385
32	Planning Time: 0.307 ms
33	Execution Time: 1176.165 ms

Total rows: 33 of 33 | Query complete 00:00:01.197

Για *max_parallel_workers_per_gather = 6* έχουμε τα εξής αποτελέσματα:

Query | Query History

```
1  SHOW max_parallel_workers_per_gather;
```

Data Output | Messages | Notifications

	max_parallel_workers_per_gather text
1	6

```
10 EXPLAIN ANALYZE
11 SELECT t::date as time_stamp, count(lon) as longtitude, count(lat) as latitude
12 FROM positions
13 GROUP BY time_stamp
14 ORDER BY longtitude desc;
```

Data Output | Messages | Notifications

	QUERY PLAN text
1	Sort (cost=676106.89..677242.92 rows=454411 width=20) (actual time=562.454..567.282 rows=24 loops=1)
2	Sort Key: (count(lon)) DESC
3	Sort Method: quicksort Memory: 26kB
4	-> Finalize GroupAggregate (cost=326617.84..624086.21 rows=454411 width=20) (actual time=562.417..567.274 rows=24 loops=1)
5	Group Key: ((t)::date)
6	-> Gather Merge (cost=326617.84..601365.66 rows=2272055 width=20) (actual time=562.412..567.254 rows=141 loops=1)
7	Workers Planned: 5
8	Workers Launched: 5
9	-> Sort (cost=325617.76..326753.79 rows=454411 width=20) (actual time=529.464..529.466 rows=24 loops=6)
10	Sort Key: ((t)::date)
11	Sort Method: quicksort Memory: 26kB
12	Worker 0: Sort Method: quicksort Memory: 26kB
13	Worker 1: Sort Method: quicksort Memory: 26kB
14	Worker 2: Sort Method: quicksort Memory: 26kB
15	Worker 3: Sort Method: quicksort Memory: 26kB
16	Worker 4: Sort Method: quicksort Memory: 26kB
17	-> Partial HashAggregate (cost=251424.80..273597.08 rows=454411 width=20) (actual time=529.175..529.439 rows=24 loops=6)
18	Group Key: (t)::date
19	Planned Partitions: 8 Batches: 1 Memory Usage: 1561kB
20	Worker 0: Batches: 1 Memory Usage: 1561kB
21	Worker 1: Batches: 1 Memory Usage: 1561kB
22	Worker 2: Batches: 1 Memory Usage: 1561kB
23	Worker 3: Batches: 1 Memory Usage: 1561kB
24	Worker 4: Batches: 1 Memory Usage: 1561kB
25	-> Parallel Seq Scan on positions (cost=0.00..146754.63 rows=1407330 width=20) (actual time=0.306..372.765 rows=1172775 loop...
26	Planning Time: 0.093 ms
27	Execution Time: 567.610 ms

Total rows: 27 of 27 Query complete 00:00:00.595

```
3  EXPLAIN ANALYZE
4  SELECT type as vessel_type, count(V.id) as vessels_with_greek_flag
5  FROM vessels AS V
6  JOIN positions AS P ON V.id = P.vessel_id
7  WHERE V.flag = 'Greece'
8  GROUP BY type;
```

Data Output | Messages | Notifications

	QUERY PLAN text
1	Finalize GroupAggregate (cost=151491.09..151512.20 rows=33 width=10) (actual time=753.264..757.529 rows=29 loops=1)
2	Group Key: v.type
3	-> Gather Merge (cost=151491.09..151511.05 rows=165 width=10) (actual time=753.257..757.502 rows=174 loops=1)
4	Workers Planned: 5
5	Workers Launched: 5
6	-> Sort (cost=150491.02..150491.10 rows=33 width=10) (actual time=704.437..704.439 rows=29 loops=6)
7	Sort Key: v.type
8	Sort Method: quicksort Memory: 26kB
9	Worker 0: Sort Method: quicksort Memory: 26kB
10	Worker 1: Sort Method: quicksort Memory: 26kB
11	Worker 2: Sort Method: quicksort Memory: 26kB
12	Worker 3: Sort Method: quicksort Memory: 26kB
13	Worker 4: Sort Method: quicksort Memory: 26kB
14	-> Partial HashAggregate (cost=150489.85..150490.18 rows=33 width=10) (actual time=704.368..704.372 rows=29 loops=6)
15	Group Key: v.type
16	Batches: 1 Memory Usage: 24kB
17	Worker 0: Batches: 1 Memory Usage: 24kB
18	Worker 1: Batches: 1 Memory Usage: 24kB
19	Worker 2: Batches: 1 Memory Usage: 24kB
20	Worker 3: Batches: 1 Memory Usage: 24kB
21	Worker 4: Batches: 1 Memory Usage: 24kB
22	-> Hash Join (cost=16.16..146978.72 rows=702226 width=67) (actual time=0.651..554.986 rows=912483 loops=6)
23	Hash Cond: ((p.vessel_id)::text = (v.id)::text)
24	-> Parallel Seq Scan on positions p (cost=0.00..143236.30 rows=1407330 width=65) (actual time=0.275..299.210 rows=1172775 loop
25	-> Hash (cost=13.11..13.11 rows=244 width=67) (actual time=0.356..0.357 rows=244 loops=6)
26	Buckets: 1024 Batches: 1 Memory Usage: 32kB
27	-> Seq Scan on vessels v (cost=0.00..13.11 rows=244 width=67) (actual time=0.220..0.309 rows=244 loops=6)
28	Filter: ((flag)::text = 'Greece'::text)
29	Rows Removed by Filter: 245
30	Planning Time: 0.299 ms
31	Execution Time: 757.566 ms

Total rows: 31 of 31 | Query complete 00:00:00.793

```
EXPLAIN ANALYZE
SELECT type, count(V.type)
FROM vessels AS V
JOIN positions AS P ON V.id = P.vessel_id
WHERE P.speed > 30
GROUP BY V.type;
```

Data Output | Messages | Notifications

	QUERY PLAN text
1	Finalize GroupAggregate (cost=148087.00..148108.11 rows=33 width=10) (actual time=445.729..450.498 rows=10 loops=1)
2	Group Key: v.type
3	-> Gather Merge (cost=148087.00..148106.95 rows=165 width=10) (actual time=445.714..450.480 rows=35 loops=1)
4	Workers Planned: 5
5	Workers Launched: 5
6	-> Sort (cost=147086.92..147087.01 rows=33 width=10) (actual time=412.700..412.702 rows=6 loops=6)
7	Sort Key: v.type
8	Sort Method: quicksort Memory: 25kB
9	Worker 0: Sort Method: quicksort Memory: 25kB
10	Worker 1: Sort Method: quicksort Memory: 25kB
11	Worker 2: Sort Method: quicksort Memory: 25kB
12	Worker 3: Sort Method: quicksort Memory: 25kB
13	Worker 4: Sort Method: quicksort Memory: 25kB
14	-> Partial HashAggregate (cost=147085.76..147086.09 rows=33 width=10) (actual time=412.658..412.661 rows=6 loops=6)
15	Group Key: v.type
16	Batches: 1 Memory Usage: 24kB
17	Worker 0: Batches: 1 Memory Usage: 24kB
18	Worker 1: Batches: 1 Memory Usage: 24kB
19	Worker 2: Batches: 1 Memory Usage: 24kB
20	Worker 3: Batches: 1 Memory Usage: 24kB
21	Worker 4: Batches: 1 Memory Usage: 24kB
22	-> Hash Join (cost=18.00..146881.04 rows=40944 width=2) (actual time=2.659..406.174 rows=34957 loops=6)
23	Hash Cond: ((p.vessel_id)::text = (v.id)::text)
24	-> Parallel Seq Scan on positions p (cost=0.00..146754.63 rows=40944 width=65) (actual time=2.275..397.011 rows=34957 loop...
25	Filter: (speed > '30'::numeric)
26	Rows Removed by Filter: 1137818
27	-> Hash (cost=11.89..11.89 rows=489 width=67) (actual time=0.360..0.360 rows=489 loops=6)
28	Buckets: 1024 Batches: 1 Memory Usage: 56kB
29	-> Seq Scan on vessels v (cost=0.00..11.89 rows=489 width=67) (actual time=0.230..0.281 rows=489 loops=6)
30	Planning Time: 0.309 ms
31	Execution Time: 450.544 ms

Total rows: 31 of 31 Query complete 00:00:00.474

```
EXPLAIN ANALYZE
SELECT P.t::date AS given_day, count(lon) AS registered_vessel_pisition
FROM positions AS P
JOIN vessels AS V on P.vessel_id = V.id
JOIN vessel_types AS VT ON  V.type = VT.code
WHERE VT.description LIKE 'Passenger,%'
    AND P.t::date BETWEEN '2019-08-14' AND '2019-08-16'
GROUP BY P.t::date;
```

Data Output | Messages | Notifications

	QUERY PLAN text
1	GroupAggregate (cost=158391.72..158851.15 rows=3319 width=12) (actual time=453.839..527.940 rows=5 loops=1)
2	Group Key: ((p.t)::date)
3	-> Gather Merge (cost=158391.72..158793.07 rows=3319 width=12) (actual time=449.444..500.818 rows=361493 loops=1)
4	Workers Planned: 5
5	Workers Launched: 5
6	-> Sort (cost=157391.64..157393.30 rows=664 width=12) (actual time=401.640..403.762 rows=60249 loops=6)
7	Sort Key: ((p.t)::date)
8	Sort Method: quicksort Memory: 3888kB
9	Worker 0: Sort Method: quicksort Memory: 3890kB
10	Worker 1: Sort Method: quicksort Memory: 3852kB
11	Worker 2: Sort Method: quicksort Memory: 3989kB
12	Worker 3: Sort Method: quicksort Memory: 3940kB
13	Worker 4: Sort Method: quicksort Memory: 3781kB
14	-> Hash Join (cost=16.24..157360.52 rows=664 width=12) (actual time=87.717..394.093 rows=60249 loops=6)
15	Hash Cond: ((p.vessel_id)::text = (v.id)::text)
16	-> Parallel Seq Scan on positions p (cost=0.00..157309.60 rows=7037 width=81) (actual time=86.262..362.325 rows=196202 loop...
17	Filter: (((t)::date >= '2019-08-14'::date) AND ((t)::date <= '2019-08-16'::date))
18	Rows Removed by Filter: 976573
19	-> Hash (cost=15.67..15.67 rows=46 width=65) (actual time=0.993..0.996 rows=64 loops=6)
20	Buckets: 1024 Batches: 1 Memory Usage: 15kB
21	-> Hash Join (cost=2.45..15.67 rows=46 width=65) (actual time=0.879..0.976 rows=64 loops=6)
22	Hash Cond: ((v.type)::text = (vt.code)::text)
23	-> Seq Scan on vessels v (cost=0.00..11.89 rows=489 width=67) (actual time=0.557..0.600 rows=489 loops=6)
24	-> Hash (cost=2.33..2.33 rows=10 width=2) (actual time=0.307..0.308 rows=10 loops=6)
25	Buckets: 1024 Batches: 1 Memory Usage: 9kB
26	-> Seq Scan on vessel_types vt (cost=0.00..2.33 rows=10 width=2) (actual time=0.290..0.295 rows=10 loops=6)
27	Filter: ((description)::text ~~ 'Passenger,%'::text)
28	Rows Removed by Filter: 96
29	Planning Time: 0.414 ms
30	Execution Time: 527.621 ms

Total rows: 30 of 30 Query complete 00:00:00.551

```
EXPLAIN ANALYZE
SELECT V.id as vessel_id, P.speed as stationary, V.type as cargo_vessels
FROM positions AS P
JOIN vessels AS V on P.vessel_id = V.id
WHERE V.type::integer BETWEEN 70 and 79
    AND P.speed = 0
    AND P.t BETWEEN '2019-08-15' AND '2019-08-18'
GROUP BY V.id, P.speed
```

Data Output | Messages | Notifications

	QUERY PLAN text
1	Group (cost=155091.15..155094.41 rows=2 width=71) (actual time=444.547..448.918 rows=38 loops=1)
2	Group Key: v.id
3	-> Gather Merge (cost=155091.15..155094.39 rows=10 width=71) (actual time=444.546..448.898 rows=174 loops=1)
4	Workers Planned: 5
5	Workers Launched: 5
6	-> Group (cost=154091.08..154092.11 rows=2 width=71) (actual time=413.031..413.426 rows=29 loops=6)
7	Group Key: v.id
8	-> Sort (cost=154091.08..154092.09 rows=406 width=71) (actual time=413.029..413.142 rows=3547 loops=6)
9	Sort Key: v.id
10	Sort Method: quicksort Memory: 391kB
11	Worker 0: Sort Method: quicksort Memory: 422kB
12	Worker 1: Sort Method: quicksort Memory: 454kB
13	Worker 2: Sort Method: quicksort Memory: 443kB
14	Worker 3: Sort Method: quicksort Memory: 431kB
15	Worker 4: Sort Method: quicksort Memory: 433kB
16	-> Hash Join (cost=19.25..154073.49 rows=406 width=71) (actual time=127.297..410.072 rows=3547 loops=6)
17	Hash Cond: ((p.vessel_id)::text = (v.id)::text)
18	-> Parallel Seq Scan on positions p (cost=0.00..153791.28 rows=99314 width=69) (actual time=112.489..400.792 rows=80102 loops=6)
19	Filter: ((t >= '2019-08-15 00:00:00'::timestamp without time zone) AND (t <= '2019-08-18 00:00:00'::timestamp without time zone) AND (speed = '0'::numeric))
20	Rows Removed by Filter: 1092673
21	-> Hash (cost=19.23..19.23 rows=2 width=67) (actual time=0.573..0.574 rows=104 loops=6)
22	Buckets: 1024 Batches: 1 Memory Usage: 19kB
23	-> Seq Scan on vessels v (cost=0.00..19.23 rows=2 width=67) (actual time=0.469..0.546 rows=104 loops=6)
24	Filter: (((type)::integer >= 70) AND ((type)::integer <= 79))
25	Rows Removed by Filter: 385
26	Planning Time: 0.312 ms
27	Execution Time: 449.001 ms

Total rows: 27 of 27 Query complete 00:00:00.484

```
SELECT P.vessel_id as completely_stationary_vessels_betweeen_12_and_18_August2019
FROM positions AS P
JOIN vessels AS V on P.vessel_id = V.id
WHERE V.type::integer BETWEEN 70 and 79
    AND P.t::date BETWEEN '2019-08-12' AND '2019-08-19'
GROUP BY P.vessel_id
HAVING MAX(P.speed) = 0;
```

Data Output | Messages | Notifications

	QUERY PLAN text
1	GroupAggregate (cost=158778.61..158833.95 rows=1 width=65) (actual time=1083.145..1152.714 rows=1 loops=1)
2	Group Key: p.vessel_id
3	Filter: (max(p.speed) = '0'::numeric)
4	Rows Removed by Filter: 52
5	-> Gather Merge (cost=158778.61..158831.43 rows=144 width=69) (actual time=958.997..1138.665 rows=94599 loops=1)
6	Workers Planned: 5
7	Workers Launched: 5
8	-> Merge Join (cost=157778.53..157814.01 rows=29 width=69) (actual time=896.712..967.055 rows=15766 loops=6)
9	Merge Cond: ((p.vessel_id)::text = (v.id)::text)
10	-> Sort (cost=157759.29..157776.89 rows=7037 width=69) (actual time=895.819..945.322 rows=261906 loops=6)
11	Sort Key: p.vessel_id
12	Sort Method: external merge Disk: 19488kB
13	Worker 0: Sort Method: external merge Disk: 22128kB
14	Worker 1: Sort Method: external merge Disk: 21096kB
15	Worker 2: Sort Method: external merge Disk: 20496kB
16	Worker 3: Sort Method: external merge Disk: 19824kB
17	Worker 4: Sort Method: external merge Disk: 20880kB
18	-> Parallel Seq Scan on positions p (cost=0.00..157309.60 rows=7037 width=69) (actual time=98.236..353.496 rows=265384 loop...
19	Filter: (((t)::date >= '2019-08-12'::date) AND ((t)::date <= '2019-08-19'::date))
20	Rows Removed by Filter: 907391
21	-> Sort (cost=19.24..19.24 rows=2 width=65) (actual time=0.886..0.902 rows=104 loops=6)
22	Sort Key: v.id
23	Sort Method: quicksort Memory: 33kB
24	Worker 0: Sort Method: quicksort Memory: 33kB
25	Worker 1: Sort Method: quicksort Memory: 33kB
26	Worker 2: Sort Method: quicksort Memory: 33kB
27	Worker 3: Sort Method: quicksort Memory: 33kB
28	Worker 4: Sort Method: quicksort Memory: 33kB
29	-> Seq Scan on vessels v (cost=0.00..19.23 rows=2 width=65) (actual time=0.520..0.608 rows=104 loops=6)
30	Filter: (((type)::integer >= 70) AND ((type)::integer <= 79))
31	Rows Removed by Filter: 385
32	Planning Time: 0.319 ms
33	Execution Time: 1155.672 ms

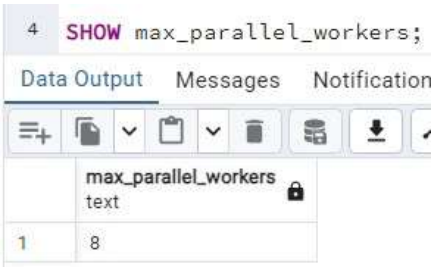
Total rows: 33 of 33 Query complete 00:00:01.197

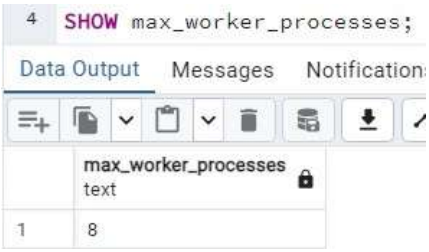
Παρατηρήσεις και συμπεράσματα

Το πράσινο χρώμα σημαίνει ότι ο χρόνος εκτέλεσης μειώθηκε από την προηγούμενη δοκιμή (στα αριστερά του), το κόκκινο χρώμα σημαίνει ότι ο χρόνος εκτέλεσης αυξήθηκε ενώ το πορτοκαλί σημαίνει ότι παρέμεινε ίδιος με πριν. Παρατηρούμε ότι κατά την αύξηση του αριθμού των workers, της PostgreSQL, ο χρόνος εκτέλεσης των queries μειώθηκε ενώ παράλληλα αυξήθηκε η απαίτηση για επεξεργαστική ισχύ κατά την εκτέλεση των queries. Πιο συγκεκριμένα, έχουμε τα εξής αποτελέσματα:

Query No. per worker's value	2 (default)	3	4	5	6
Query 1	834ms	708ms	669ms	633ms	559ms
Query 2	1090ms	911ms	837ms	771ms	757ms
Query 3	659ms	524ms	504ms	512ms	450ms
Query 4	664ms	596ms	569ms	563ms	527ms
Query 5A	564ms	469ms	453ms	444ms	449ms
Query 5B	1877ms	1421ms	1273ms	1176ms	1155ms

Επίσης, διαπιστώνουμε μελετώντας την εκτύπωση της *EXPLAIN ANALYZE* **ότι κανένα από τα 6 queries δεν χρησιμοποίησε πάνω από 5 workers**. Επιπλέον, σε δοκιμές που έγιναν με 12 και 64 workers το CPU load έφτασε έως και το 65-70%, η θερμοκρασία του επεξεργαστή αυξήθηκε απότομα και το αποτέλεσμα σε χρόνους ήταν το ίδιο ή χειρότερο από τα παραπάνω. Μια πρώτη προσέγγιση στο γιατί χρησιμοποιήθηκαν λιγότεροι workers από αυτούς που ορίσαμε με την εντολή *max_parallel_workers_per_gather* μας οδήγησε στις παραμέτρους *max_worker_processes* και *max_parallel_workers*. Οι δύο αυτές παράμετροι έχουν default όριο το 8 όπως βλέπουμε παρακάτω:





Ωστόσο, η τιμή "*Workers Planned*" σε όλα τα παραπάνω πλάνα εκτέλεσης ποτέ δεν ξεπέρασε την τιμή των "Workers Launched" που σημαίνει ότι η PostgreSQL δεν περιορίστηκε από το λειτουργικό στην χρήση παραπάνω επεξεργαστικών πόρων. Το βέλτιστο μονοπάτι για την απάντηση του κάθε ερωτήματος βρέθηκε με λιγότερους workers από ότι παραχωρήθηκαν στη PostgreSQL και έτσι στην περίπτωση των 64 max_parallel_workers_per_gather είχαμε και πάλι 'Workers Planned = 5' και 'Workers_Launched = 5'. Για παράδειγμα, στην περίπτωση που είχαμε 'Workers Planned' = 9 και 'Workers Launched' = 7 τότε θα έπρεπε να βεβαιωθούμε πως δεν περιορίζεται η PostgreSQL μέσω αυτών των δύο παραμέτρων, ορίζοντας της ανάλογες τιμές και τρέχοντας εκ νέου τις ερωτήσεις μας για τυχόν βελτιστοποίησης στους χρόνους εκτέλεσης.

Ερώτημα 4ο – Indexing

Η γενική ιδέα των ερυετηρίων που θα ακολουθήσουμε είναι να μειώσουμε τους χρόνους εκτέλεσης των ερωτήσεων στοχεύοντας κυρίως στις συνθήκες (*WHERE clause)* του κάθε ερωτήματος, όπου αυτό είναι εφικτό. Επίσης, γνωρίζουμε ότι μερικά από τα αρνητικά των ευρετηρίων είναι η αύξηση του όγκου των δεδομένων και πολλές φορές η επιβάρυνση στις εντολές *INSERT*, *DELETE* και *UPDATE* καθώς πρέπει όχι μόνο να αλλάξουν τιμές στο εκάστοτε table αλλά και οι τιμές στο ευρετήριο του.

Για την εύρεση των ευρετηρίων που έχουν δημιουργηθεί σε κάθε table της βάσης δεδομένων θα χρησιμοποιήσουμε την παρακάτω ερώτηση:

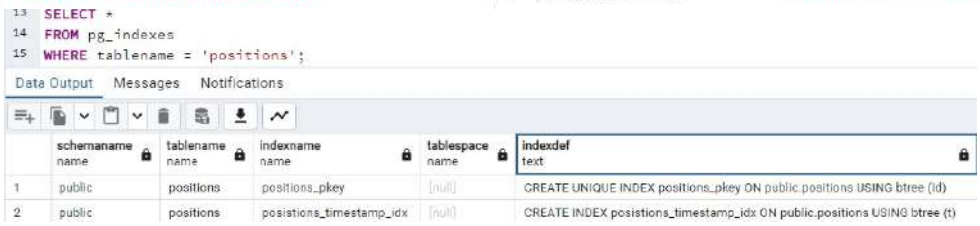
```
SELECT * FROM pg_indexes WHERE tablename = 'table_name';
```

Για την εκκαθάριση της μνήμης cache θα χρησιμοποιηθεί η εντολή `VACUUM FULL` ενώ για την μελέτη και αξιολόγηση των ευρετηρίων η εντολή `EXPLAIN ANALYZE`, όπως ακριβώς έγινε και στα προηγούμενα ερωτήματα της εργασίας.

Δημιουργία ευρετηρίου για Query No.1

Αρχικά, παρατηρούμε ότι το συγκεκριμένο ερώτημα είναι το μόνο από τα 6 το οποίο δεν χρησιμοποιεί κάποιο συνθήκη `WHERE`. Επιλέγουμε την κατασκευή ευρετηρίου στη στήλη `positions.t` καθώς αυτή είναι η στήλη για την οποία ζητάμε ομαδοποίηση των στοιχείων στο ερώτημα μας μέσω του `GROUP BY` clause:

```
CREATE INDEX posistions_timestamp_idx ON positions(t);
```



Πράγματι, παρατηρούμε ότι ο χρόνος εκτέλεσης είναι πιο γρήγορος σε σχέση με πριν κατά 93ms:

Η συγκεκριμένη βελτίωση είναι συγκριτικά μικρότερη από αυτή που θα περιμέναμε με την χρήση ευρετηρίων διότι η ερώτηση αποτελείται από ένα μόνο πίνακα (*positions*) και όχι σύνδεσης δύο ή παραπάνω πινάκων. Επίσης, η έλλειψη συνθήκης αναζήτησης συγκεκριμένου στοιχείου (πέραν του πρωτεύοντος κλειδιού: id) π.χ. συγκεκριμένης ημερομηνίας: `WHERE positions.t = '2019-8-14'` κάνει εξαρχής το ερώτημα στη βάση δεδομένων πιο αργό αφού πρέπει να προσπελάσει ολόκληρο τον πίνακα για την ταξινόμηση `lon DESC`. Τέλος, ο σημαντικότερο λόγος που δεν είδαμε κάποια πραγματική βελτίωση στους χρόνους εκτέλεσης του παραπάνω ερωτήματος είναι **διότι δεν χρησιμοποιείται το ευρετήριο που κατασκευάσαμε καθώς η ημερομηνία που ζητάμε πρέπει να γίνει *cast* σε date format** (`P.t::date BETWEEN '2019-08-14' AND '2019-08-18'`) και αυτό γίνεται για κάθε ημερομηνία στην στήλη `positions.t`. Θα μπορούσε να αποφευχθεί το cast και να γίνει μια πιο σύνθετη συνθήκη που θα περιλαμβάνει και τα timestamps των ημερομηνιών που ζητούνται με σκοπό το ευρετήριο να χρησιμοποιηθεί από τον Query Optimizer της PostgreSQL.

Δημιουργία ευρετηρίου για Query No.2

Αρχικά, εκτελούμε την εντολή εκκαθάρισης της μνήμης cache:

```
VACUUM FULL
```

Στην συνέχεια, επιλέγουμε την κατασκευή ευρετηρίου τύπου Hash, στη στήλη *vessels.flag* καθώς αποτελεί την συνθήκη WHERE με ισότητά, του ερωτήματος μας (`WHERE vessels.flag = 'Greece'`) και με αυτό τον τρόπο θα ελαχιστοποιήσουμε τους χρόνους αναζήτησης. Η βελτίωση που θα δούμε στους χρόνους εκτέλεσης οφείλεται στο ότι στην

	QUERY PLAN text
1	Finalize GroupAggregate (cost=174577.63..174585.99 rows=33 width=10) (actual time=1138.480..1142.563 rows=29 loops=1)
2	Group Key: v.type
3	-> Gather Merge (cost=174577.63..174585.33 rows=66 width=10) (actual time=1138.469..1142.544 rows=87 loops=1)
4	Workers Planned: 2
5	Workers Launched: 2
6	-> Sort (cost=173577.60..173577.68 rows=33 width=10) (actual time=1118.413..1118.416 rows=29 loops=3)
7	Sort Key: v.type
8	Sort Method: quicksort Memory: 26kB
9	Worker 0: Sort Method: quicksort Memory: 26kB
10	Worker 1: Sort Method: quicksort Memory: 26kB
11	-> Partial HashAggregate (cost=173576.44..173576.77 rows=33 width=10) (actual time=1118.346..1118.351 rows=29 loops=3)
12	Group Key: v.type
13	Batches: 1 Memory Usage: 24kB
14	Worker 0: Batches: 1 Memory Usage: 24kB
15	Worker 1: Batches: 1 Memory Usage: 24kB
16	-> Hash Join (cost=16.16..166261.58 rows=1462971 width=67) (actual time=0.523..833.529 rows=1824966 loops=3)
17	Hash Cond: ((p.vessel_id)::text = (v.id)::text)
18	-> Parallel Seq Scan on positions p (cost=0.00..158482.38 rows=2931938 width=65) (actual time=0.200..348.645 rows=2345550 loop...
19	-> Hash (cost=13.11..13.11 rows=244 width=67) (actual time=0.303..0.303 rows=244 loops=3)
20	Buckets: 1024 Batches: 1 Memory Usage: 32kB
21	-> Seq Scan on vessels v (cost=0.00..13.11 rows=244 width=67) (actual time=0.192..0.260 rows=244 loops=3)
22	Filter: ((flag)::text = 'Greece'::text)
23	Rows Removed by Filter: 245
24	Planning Time: 0.376 ms
25	Execution Time: 1142.616 ms

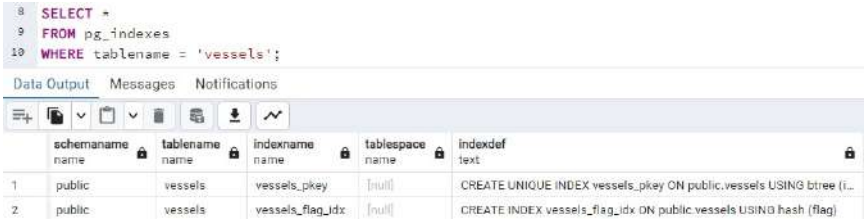
Ευρετήριο στη στήλη vessels.flag

	QUERY PLAN text
1	Finalize GroupAggregate (cost=174577.63..174585.99 rows=33 width=10) (actual time=1218.120..1221.964 rows=29 loops=1)
2	Group Key: v.type
3	-> Gather Merge (cost=174577.63..174585.33 rows=66 width=10) (actual time=1218.114..1221.943 rows=87 loops=1)
4	Workers Planned: 2
5	Workers Launched: 2
6	-> Sort (cost=173577.60..173577.68 rows=33 width=10) (actual time=1198.690..1198.692 rows=29 loops=3)
7	Sort Key: v.type
8	Sort Method: quicksort Memory: 26kB
9	Worker 0: Sort Method: quicksort Memory: 26kB
10	Worker 1: Sort Method: quicksort Memory: 26kB
11	-> Partial HashAggregate (cost=173576.44..173576.77 rows=33 width=10) (actual time=1198.621..1198.626 rows=29 loops=3)
12	Group Key: v.type
13	Batches: 1 Memory Usage: 24kB
14	Worker 0: Batches: 1 Memory Usage: 24kB
15	Worker 1: Batches: 1 Memory Usage: 24kB
16	-> Hash Join (cost=16.16..166261.58 rows=1462971 width=67) (actual time=0.498..895.939 rows=1824966 loops=3)
17	Hash Cond: ((p.vessel_id)::text = (v.id)::text)
18	-> Parallel Seq Scan on positions p (cost=0.00..158482.38 rows=2931938 width=65) (actual time=0.199..384.841 rows=2345550 loop...
19	-> Hash (cost=13.11..13.11 rows=244 width=67) (actual time=0.282..0.283 rows=244 loops=3)
20	Buckets: 1024 Batches: 1 Memory Usage: 32kB
21	-> Seq Scan on vessels v (cost=0.00..13.11 rows=244 width=67) (actual time=0.179..0.242 rows=244 loops=3)
22	Filter: ((flag)::text = 'Greece'::text)
23	Rows Removed by Filter: 245
24	Planning Time: 0.278 ms
25	Execution Time: 1222.010 ms

Χωρίς χρήση ευρετηρίου

αρχική περίπτωση η PostgreSQL ψάχνει σειριακά την στήλη *flag*, ενώ τώρα η αναζήτηση θα γίνει με την βοήθεια Hash για την σημαία του κάθε πλοίου:

```
CREATE INDEX vessels_flag_idx ON vessels USING HASH (flag);
```



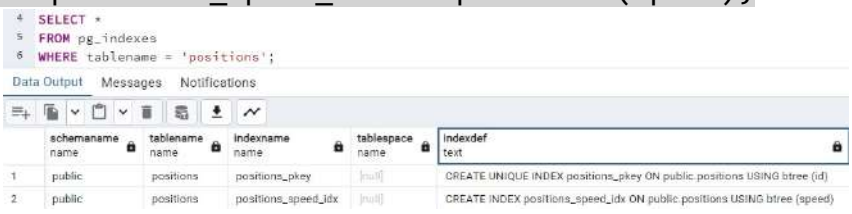
Πράγματι, παρατηρούμε ότι ο χρόνος εκτέλεσης είναι πιο γρήγορος σε σχέση με πριν κατά 80ms:

Γενικότερα, τα hash indexes είναι κατάλληλα όταν ψάχνουμε συνθήκη με ισότητα πάνω σε κάποιο πίνακα και αυτό είναι το μόνο από τα έξι επερώτημα στην βάση μας που πληροί τις συνθήκες αυτές.

Δημιουργία ευρετηρίου για Query No.3

Στο συγκεκριμένο επερώτημα παρατηρούμε ότι γίνεται χρήση συνθήκης με συγκριτικό τελεστή `>`. Για τη βελτίωση του χρόνου εκτέλεσης του θα χρησιμοποιήσουμε ευρετηρίου τύπου B-tree στην στήλη `positions.speed`.

```
CREATE INDEX positions_speed_idx ON positions(speed);
```



Πράγματι, παρατηρούμε ότι ο χρόνος εκτέλεσης είναι πιο γρήγορος σε σχέση με πριν κατά 441ms:

	QUERY PLAN text
1	Finalize GroupAggregate (cost=166602.30..166611.66 rows=33 width=10) (actual time=174.861..193.352 rows=10 loops=1)
2	Group Key: v.type
3	-> Gather Merge (cost=166603.30..166611.00 rows=66 width=10) (actual time=174.857..183.343 rows=28 loops=1)
4	Workers Planned: 2
5	Workers Launched: 2
6	-> Sort (cost=165603.28..165603.36 rows=33 width=10) (actual time=150.551..150.552 rows=9 loops=3)
7	Sort Key: v.type
8	Sort Method: quicksort Memory: 25kB
9	Worker 0: Sort Method: quicksort Memory: 25kB
10	Worker 1: Sort Method: quicksort Memory: 25kB
11	-> Partial HashAggregate (cost=165602.12..165602.45 rows=33 width=10) (actual time=150.514..150.516 rows=9 loops=3)
12	Group Key: v.type
13	Batches: 1 Memory Usage: 24kB
14	Worker 0: Batches: 1 Memory Usage: 24kB
15	Worker 1: Batches: 1 Memory Usage: 24kB
16	-> Hash Join (cost=3857.00..165175.62 rows=85299 width=2) (actual time=9.689..139.105 rows=69914 loops=3)
17	Hash Cond: ((p.vessel_id)::text = (v.id)::text)
18	-> Parallel Bitmap Heap Scan on positions p (cost=3839.00..164931.77 rows=85299 width=65) (actual time=9.218..121.577 rows=69914 loo...
19	Recheck Cond: (speed > '30'::numeric)
20	Heap Blocks: exact=14072
21	-> Bitmap Index Scan on positions_speed_idx (cost=0.00..3787.82 rows=204718 width=0) (actual time=22.992..22.992 rows=209741 loo...
22	Index Cond: (speed > '30'::numeric)
23	-> Hash (cost=11.89..11.89 rows=489 width=67) (actual time=0.450..0.451 rows=489 loops=3)
24	Buckets: 1024 Batches: 1 Memory Usage: 56kB
25	-> Seq Scan on vessels v (cost=0.00..11.89 rows=489 width=67) (actual time=0.302..0.359 rows=489 loops=3)
26	Planning Time: 0.353 ms
27	Execution Time: 183.403 ms

Ευρετήριο στη στήλη positions.speed

	QUERY PLAN text
1	Finalize GroupAggregate (cost=167483.75..167492.11 rows=33 width=10) (actual time=620.855..624.300 rows=10 loops=1)
2	Group Key: v.type
3	-> Gather Merge (cost=167483.75..167491.45 rows=66 width=10) (actual time=620.847..624.289 rows=22 loops=1)
4	Workers Planned: 2
5	Workers Launched: 2
6	-> Sort (cost=166483.73..166483.81 rows=33 width=10) (actual time=602.582..602.585 rows=7 loops=3)
7	Sort Key: v.type
8	Sort Method: quicksort Memory: 25kB
9	Worker 0: Sort Method: quicksort Memory: 25kB
10	Worker 1: Sort Method: quicksort Memory: 25kB
11	-> Partial HashAggregate (cost=166482.57..166482.90 rows=33 width=10) (actual time=602.538..602.541 rows=7 loops=3)
12	Group Key: v.type
13	Batches: 1 Memory Usage: 24kB
14	Worker 0: Batches: 1 Memory Usage: 24kB
15	Worker 1: Batches: 1 Memory Usage: 24kB
16	-> Hash Join (cost=18.00..166056.07 rows=85299 width=2) (actual time=0.932..590.878 rows=69914 loops=3)
17	Hash Cond: ((p.vessel_id)::text = (v.id)::text)
18	-> Parallel Seq Scan on positions p (cost=0.00..165812.22 rows=85299 width=65) (actual time=0.449..573.623 rows=69914 loop
19	Filter: (speed > '30'::numeric)
20	Rows Removed by Filter: 2275637
21	-> Hash (cost=11.89..11.89 rows=489 width=67) (actual time=0.465..0.465 rows=489 loops=3)
22	Buckets: 1024 Batches: 1 Memory Usage: 56kB
23	-> Seq Scan on vessels v (cost=0.00..11.89 rows=489 width=67) (actual time=0.324..0.381 rows=489 loops=3)
24	Planning Time: 0.303 ms
25	Execution Time: 624.338 ms

Χωρίς χρήση ευρετηρίου

Παρατηρούμε στα αριστερά, γραμμή 21: `Bitmap Index Scan in positions_speed_idx`. Αυτό σημαίνει ότι ο Query Optimizer αποφάσισε πως αυτό είναι το βέλτιστο μονοπάτι εκτέλεσης της ερώτησης και έτσι επέλεξε την χρήση το ευρετηρίου που μόλις κατασκευάσαμε.

Γενικότερα, η χρήση των B-δένδρων μας βοηθάει στην αναζήτηση εγγραφών όταν έχουμε στο επερώτημα μας συγκριτικούς τελεστές `>` , `<` , `>=` , `<=`, `=`, διάστημα τιμών με την χρήση του `BETWEEN` keyword καθώς και regular expression (χειρότερη περίπτωση αναζήτησης) με το keyword `LIKE` “%...” όπως θα δούμε παρακάτω.

Δημιουργία ευρετηρίου για Query No.4

Αρχικά, παρατηρούμε ότι εδώ γίνεται μια επιλογή σε διάστημα με την χρήση του BETWEEN πιο συγκεκριμένα: `WHERE V.type BETWEEN 70 and 79`. Οι στήλες `positions.speed` και `positions.date` έχουν ήδη ευρετηριαστεί σε προηγούμενα ερωτήματα. Για αυτό τον λόγο αποφασίζουμε να κατασκευάσουμε ευρετήριο τύπου B-δένδρο πάνω στην στη στήλη `vessels.type`:

```
CREATE INDEX vessel_type_idx ON vessels(type);
```

Πράγματι, παρατηρούμε ότι ο χρόνος εκτέλεσης είναι πιο μικρός κατά 29ms σε σχέση με την αρχική εκτέλεση, του πρώτου ερωτήματος, ωστόσο το ευρετήριο δεν χρησιμοποιείται στο συγκεκριμένο επερώτημα:

```
EXPLAIN ANALYZE
SELECT P.t::date AS given_day, count(lon) AS registered_vessel_pisition
FROM positions AS P
JOIN vessels AS V on P.vessel_id = V.id
JOIN vessel_types AS VT ON  V.type = VT.code
WHERE VT.description LIKE 'Passenger,%'
    AND P.t::date BETWEEN '2019-08-14' AND '2019-08-18'
GROUP BY P.t::date;
```

Data Output | Messages | Notifications

```
QUERY PLAN
text
Finalize GroupAggregate  (cost=188962.40..189364.64 rows=3319 width=12) (actual time=592.622..609.329 rows=5 loops=1)
  Group Key: ((p.t)::date)
  -> Gather Merge  (cost=188962.40..189309.32 rows=2766 width=12) (actual time=588.396..609.308 rows=15 loops=1)
        Workers Planned: 2
        Workers Launched: 2
        -> Partial GroupAggregate  (cost=187962.37..187990.03 rows=1383 width=12) (actual time=550.787..567.968 rows=5 loops=3)
              Group Key: ((p.t)::date)
              -> Sort  (cost=187962.37..187965.83 rows=1383 width=12) (actual time=549.607..558.241 rows=120498 loops=3)
                    Sort Key: ((p.t)::date)
                    Sort Method: external merge  Disk: 3040kB
                    Worker 0:  Sort Method: external merge  Disk: 2520kB
                    Worker 1:  Sort Method: external merge  Disk: 2512kB
                    -> Hash Join  (cost=16.24..187890.22 rows=1383 width=12) (actual time=0.716..529.890 rows=120498 loops=3)
                          Hash Cond: ((p.vessel_id)::text = (v.id)::text)
                          -> Parallel Seq Scan on positions p  (cost=0.00..187801.76 rows=14660 width=81) (actual time=0.212..472.063 rows=392404 loops=3)
                                Filter: (((t)::date >= '2019-08-14'::date) AND ((t)::date <= '2019-08-18'::date))
                                Rows Removed by Filter: 1953146
                          -> Hash  (cost=15.67..15.67 rows=46 width=65) (actual time=0.468..0.470 rows=64 loops=3)
                                Buckets: 1024  Batches: 1  Memory Usage: 15kB
                                -> Hash Join  (cost=2.45..15.67 rows=46 width=65) (actual time=0.380..0.458 rows=64 loops=3)
                                      Hash Cond: ((v.type)::text = (vt.code)::text)
                                      -> Seq Scan on vessels v  (cost=0.00..11.89 rows=489 width=67) (actual time=0.199..0.233 rows=489 loops=3)
                                      -> Hash  (cost=2.33..2.33 rows=10 width=2) (actual time=0.170..0.170 rows=10 loops=3)
                                            Buckets: 1024  Batches: 1  Memory Usage: 9kB
                                            -> Seq Scan on vessel_types vt  (cost=0.00..2.33 rows=10 width=2) (actual time=0.158..0.162 rows=10 loops=3)
                                                  Filter: ((description)::text ~~ 'Passenger,%'::text)
                                                  Rows Removed by Filter: 96
Planning Time: 0.449 ms
Execution Time: 610.594 ms
```

Ο λόγος που δεν είδαμε κάποια μεγάλη βελτίωση στους χρόνους εκτέλεσης του παραπάνω ερωτήματος είναι **διότι δεν χρησιμοποιείται το ευρετήριο που κατασκευάσαμε καθώς η ημερομηνία που ζητάμε πρέπει να γίνει *cast* σε `date` format** (`P.t::date BETWEEN '2019-08-14' AND '2019-08-18'`) και αυτό γίνεται για κάθε ημερομηνία στην στήλη `positions.t` όπως ακριβώς και στο *Query No.1*. Θα μπορούσε να αποφευχθεί το cast και να γίνει μια πιο σύνθετη συνθήκη που θα περιλαμβάνει και τα timestamps των ημερομηνιών που ζητούνται με σκοπό το ευρετήριο να χρησιμοποιηθεί από τον Query Optimizer της PostgreSQL.

Δημιουργία ευρετηρίου για Query No.5A

Μια πρώτη βελτίωση που παρατηρούμε στο ερώτημα μας είναι να αποφύγουμε το type casting και αντί για `V.type::integer BETWEEN 70 and 79` μπορούμε να γράψουμε απλός `V.type BETWEEN '70' and '79'` μιας και η στήλη `vessels.type` είναι δηλωμένη ως `VARCHAR(3)`.

```
EXPLAIN ANALYZE
SELECT V.id as vessel_id, P.speed as stationary, V.type as cargo_vessels
FROM positions AS P
JOIN vessels AS V on P.vessel_id = V.id
WHERE V.type BETWEEN '70' and '79'
    AND P.speed = 0
    AND P.t BETWEEN '2019-08-15' AND '2019-08-18'
GROUP BY V.id, P.speed
```

Data Output | Messages | Notifications

	QUERY PLAN text
1	Group (cost=182151.06..182176.09 rows=105 width=71) (actual time=516.536..520.561 rows=30 loops=1)
2	Group Key: v.id
3	-> Gather Merge (cost=182151.06..182175.56 rows=210 width=71) (actual time=516.535..520.547 rows=89 loops=1)
4	Workers Planned: 2
5	Workers Launched: 2
6	-> Sort (cost=181151.04..181151.30 rows=105 width=71) (actual time=496.948..496.945 rows=30 loops=3)
7	Sort Key: v.id
8	Sort Method: quicksort Memory: 27kB
9	Worker 0: Sort Method: quicksort Memory: 27kB
10	Worker 1: Sort Method: quicksort Memory: 27kB
11	-> Partial HashAggregate (cost=181146.46..181147.51 rows=105 width=71) (actual time=496.867..496.873 rows=30 loops=3)
12	Group Key: v.id
13	Batches: 1 Memory Usage: 24kB
14	Worker 0: Batches: 1 Memory Usage: 24kB
15	Worker 1: Batches: 1 Memory Usage: 24kB
16	-> Hash Join (cost=15.65..181035.39 rows=44428 width=71) (actual time=153.543..495.449 rows=7094 loops=3)
17	Hash Cond: ((p.vessel_id)::text = (v.id)::text)
18	-> Parallel Seq Scan on positions p (cost=0.00..180471.91 rows=206905 width=69) (actual time=122.852..479.849 rows=160205 loops=3)
19	Filter: ((t >= '2019-08-15 00:00:00'::timestamp without time zone) AND (t <= '2019-08-18 00:00:00'::timestamp without time zone) AND (speed = '0'::num...
20	Rows Removed by Filter: 2185346
21	-> Hash (cost=14.34..14.34 rows=105 width=67) (actual time=0.516..0.516 rows=104 loops=3)
22	Buckets: 1024 Batches: 1 Memory Usage: 19kB
23	-> Seq Scan on vessels v (cost=0.00..14.34 rows=105 width=67) (actual time=0.330..0.495 rows=104 loops=3)
24	Filter: (((type)::text >= '70'::text) AND ((type)::text <= '79'::text))
25	Rows Removed by Filter: 385
26	Planning Time: 0.442 ms
27	Execution Time: 520.604 ms

Δημιουργία ευρετηρίου για Query No.5B

Δοκιμάστηκε αρχικά η κατασκευή διαφόρων ευρετηρίων τα οποία είτε έπαιρναν πάνω από 30’ να υλοποιηθούν, μεταξύ άλλων `create index position_speed_idx on positions USING HASH (speed) where speed = 0;`,

```
30  create index position_speed_idx on positions USING HASH (speed) where speed = 0
```

Data Output | Messages | Notifications

```
CREATE INDEX

Query returned successfully in 38 min 52 secs.
```

καθώς υπάρχουν 7 εκ. εγγραφές στον πίνακα positions, και ευρετήρια τύπου B-tree τα οποία δεν χρησιμοποιήθηκαν ποτέ από τον *query optimizer* καθώς ήταν πιο αργά μονοπάτια από την γραμμική αναζήτηση. Ωστόσο, καταφέραμε να υλοποιήσουμε ευρετήριο το οποίο τελικά χρησιμοποιήθηκε και ήταν το παρακάτω:

`CREATE INDEX positions_vesselid_speed_timestamp_idx ON positions (vessel_id, speed, t)`(3)

Μερικές ακόμη αλλαγές στο συγκεκριμένο query που έγιναν για περαιτέρω βελτίωση ήταν η απαλοιφή του `BETWEEN` στην επιλογή ημερομηνιών και στην θέση του η χρήση συγκριτικών τελεστών `>=` και `<=`(1) καθώς και η αναφορά στο speed *numeric*, δηλαδή από `0` → `0.0`(2). Πράγματι, παρατηρούμε ότι ο χρόνος εκτέλεσης είναι πιο γρήγορος σε σχέση με τις αρχικές μετρήσεις κατά **1663ms** και η τελική μορφή του query πλέον είναι η παρακάτω:

```
EXPLAIN ANALYZE
SELECT P.vessel_id as completely_stationary_vessels_between_12_and_18_August2019
FROM positions AS P
JOIN vessels AS V on P.vessel_id = V.id
WHERE V.type::integer BETWEEN 70 and 79
    AND P.t >= '2019-08-12 00:00:00' AND P.t <= '2019-08-19 00:00:00'
GROUP BY P.vessel_id
HAVING MAX(P.speed) = 0.0;
```

1

2

Data Output | Messages | Notifications

	QUERY PLAN text
1	GroupAggregate (cost=1068.15..26499.95 rows=2 width=65) (actual time=199.546..288.790 rows=1 loops=1)
2	Group Key: p.vessel_id
3	Filter: (max(p.speed) = 0.0)
4	Rows Removed by Filter: 48
5	-> Nested Loop (cost=1068.15..26464.30 rows=6074 width=69) (actual time=0.537..272.651 rows=88842 loops=1)
6	-> Index Scan using vessels_pkey on vessels v (cost=0.27..74.84 rows=2 width=65) (actual time=0.008..0.472 rows=104 loops=1)
7	Filter: (((type)::integer >= 70) AND ((type)::integer <= 79))
8	Rows Removed by Filter: 385
9	-> Bitmap Heap Scan on positions p (cost=1067.87..13159.54 rows=3519 width=69) (actual time=0.779..2.508 rows=854 loops=104)
10	Recheck Cond: (((v.id)::text = (vessel_id)::text) AND (t >= '2019-08-12 00:00:00'::timestamp without time zone) AND (t <= '2019-08-19 00:00:00'::timestamp without time z...
11	Heap Blocks: exact=62035
12	-> Bitmap Index Scan on idx_positions_vessel_speed_t (cost=0.00..1066.99 rows=3519 width=0) (actual time=0.722..0.722 rows=854 loops=104)
13	Index Cond: (((vessel_id)::text = (v.id)::text) AND (t >= '2019-08-12 00:00:00'::timestamp without time zone) AND (t <= '2019-08-19 00:00:00'::timestamp without time ...
14	Planning Time: 0.460 ms
15	Execution Time: 288.825 ms

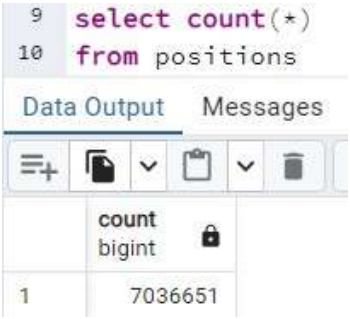
3

4

Τέλος, παρατηρούμε ότι χρησιμοποιήθηκε και 2ο ευρετήριο, το default ευρετήριο(4) της στήλης *vessels.id* με όνομα *vessels_pkey* το οποίο βοηθάει στην μείωση του χρόνου αναζήτησης όλων τω πλοίων που ικανοποιούν την συνθήκη (cargo vessel) ενώ στην αρχική υλοποίηση του ερωτήματος γινόταν σειριακά (seq scan).

Ερώτημα 5ο - Partitioning

Όπως γνωρίζουμε, η τεχνική του partitioning αναφέρεται κυρίως σε «μεγάλους» πίνακες. Στην περίπτωση μας αυτομάτως στρέφουμε την προσοχή μας στον πίνακα *positions* ο οποίος περιέχει 7036651 εγγραφές.



Επομένως, «σπάμε» τον παραπάνω πίνακα, ισομερώς, με βάση κάποιου χαρακτηριστικού. Αυτό το χαρακτηριστικό στην περίπτωση μας θα είναι η στήλη **t**, δηλαδή η ημερομηνία, με την χρήση **Declarative Partitioning** και πιο συγκεκριμένα **Range Partitioning**. Ακολουθώντας τα βήματα του official PostgreSQL documentation, για τον ορισμό partitions, στο πίνακα Positions θα χρειαστεί:

1. Να διαγράψουμε τον πίνακα που χρησιμοποιήσαμε από το 1ο ερώτημα έως τώρα και να τον αρχικοποιήσουμε ως εξής:

```
CREATE TABLE positions
(   id VARCHAR(8),
    vessel_id VARCHAR(64) NOT NULL,
t TIMESTAMP NOT NULL,     lon
NUMERIC(7,5) NOT NULL,     lat
NUMERIC(7,5) NOT NULL,     heading
NUMERIC(3),     course
NUMERIC(4,1),     speed
NUMERIC(4,1),
    FOREIGN KEY (vessel_id) REFERENCES vessels(id),
    PRIMARY KEY (id, t)
) PARTITION BY RANGE(t);
```

2. Στη συνέχεια, να ορίσουμε τα partitions ώστε κάθε ένα να αντιπροσωπεύει μια εβδομάδα του μήνα:

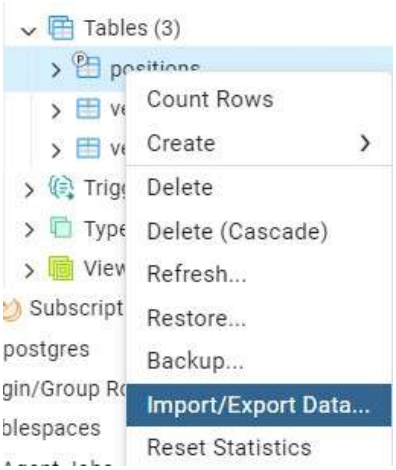
```
CREATE TABLE augustweek1 PARTITION of positions FOR VALUES FROM (MINVALUE) TO
```

```
(timestamp '2019-8-08 00:00:00');
CREATE TABLE augustweek2 PARTITION of positions FOR VALUES FROM (timestamp '2019-
8-08 00:00:00') TO (timestamp '2019-8-15 00:00:00'); CREATE TABLE augustweek3
PARTITION of positions FOR VALUES FROM (timestamp '2019-
8-15 00:00:00') TO (timestamp '2019-8-22 00:00:00'); CREATE TABLE augustweek4
PARTITION of positions FOR VALUES FROM (timestamp '2019-
8-22 00:00:00') TO (MAXVALUE);
```

3. Επόμενο βήμα είναι να γεμίσουμε τους πίνακες με τα δεδομένα εκ νέου το οποίο το πετυχαίνουμε με τις παρακάτω εντολές (ίδιες με το ερώτημα 1ο):

```
COPY vessel_types FROM '/Library/PostgreSQL/16/share/extension/vessel_types.csv'
DELIMITER ',' CSV HEADER;
COPY vessels FROM '/Library/PostgreSQL/16/share/extension/vessels.csv' DELIMITER
',' CSV HEADER;
COPY positions FROM '/Library/PostgreSQL/16/share/extension/positions.csv'
DELIMITER ',' CSV HEADER;
```

Στην περίπτωση των Windows 11 έγινε χρήση του GUI pgAdmin με την επιλογή ανά πίνακα *"Import/Export Data…"*:



4. Δημιουργία ενός ευρετηρίου πάνω στον πίνακα positions στο χαρακτηριστικό που χρησιμοποιήσαμε για το partitioning, δηλαδή πάνω στην στήλη **t**:

```
CREATE INDEX ON positions (t);
```

5. Τέλος, βεβαιωνόμαστε ότι το χαρακτηριστικό της PostgreSQL: **`enable_partition_pruning`** είναι ενεργοποιημένο (On) καθώς θα βοηθήσει στο query optimization που επιζητάμε μιας και κάνουμε χρήση declarative partitioning.

   Πράγματι, by default βλέπουμε πως είναι:

Σε περίπτωση που είναι απενεργοποιημένο, εκτελούμε την παρακάτω εντολή για την ενεργοποίηση του:

Τα αποτελέσματα των 6 queries με την χρήση Partitioning είναι τα παρακάτω:

```
EXPLAIN ANALYZE
SELECT t::date as time_stamp, count(lon) as longtitude, count(lat) as latitude
FROM positions
GROUP BY time_stamp
ORDER BY longtitude desc;
```

Data Output | Messages | Notifications

	QUERY PLAN text
1	Sort (cost=203531.43..203531.93 rows=200 width=20) (actual time=1025.553..1029.451 rows=24 loops=1)
2	Sort Key: (count(positions.lon)) DESC
3	Sort Method: quicksort Memory: 26kB
4	-> Finalize GroupAggregate (cost=203471.62..203523.79 rows=200 width=20) (actual time=1025.535..1029.443 rows=24 loops=1)
5	Group Key: ((positions.t)::date)
6	-> Gather Merge (cost=203471.62..203518.29 rows=400 width=20) (actual time=1025.532..1029.432 rows=38 loops=1)
7	Workers Planned: 2
8	Workers Launched: 2
9	-> Sort (cost=202471.59..202472.09 rows=200 width=20) (actual time=1005.623..1005.625 rows=13 loops=3)
10	Sort Key: ((positions.t)::date)
11	Sort Method: quicksort Memory: 25kB
12	Worker 0: Sort Method: quicksort Memory: 25kB
13	Worker 1: Sort Method: quicksort Memory: 25kB
14	-> Partial HashAggregate (cost=202461.45..202463.95 rows=200 width=20) (actual time=1005.606..1005.609 rows=13 loops=3)
15	Group Key: ((positions.t)::date)
16	Batches: 1 Memory Usage: 40kB
17	Worker 0: Batches: 1 Memory Usage: 40kB
18	Worker 1: Batches: 1 Memory Usage: 40kB
19	-> Parallel Append (cost=0.00..180471.91 rows=2931938 width=20) (actual time=0.178..680.068 rows=2345550 loops=3)
20	-> Parallel Seq Scan on augustweek1 positions_1 (cost=0.00..68681.93 rows=1222635 width=20) (actual time=0.305..237.355 rows=978108 loops=3)
21	-> Parallel Seq Scan on augustweek2 positions_2 (cost=0.00..37920.31 rows=666665 width=20) (actual time=0.240..380.680 rows=1599996 loops=1)
22	-> Parallel Seq Scan on augustweek4 positions_4 (cost=0.00..30082.23 rows=529938 width=20) (actual time=0.311..161.930 rows=635926 loops=2)
23	-> Parallel Seq Scan on augustweek3 positions_3 (cost=0.00..29127.75 rows=512700 width=20) (actual time=0.027..278.175 rows=1230480 loops=1)
24	Planning Time: 0.156 ms
25	Execution Time: 1029.497 ms

```
EXPLAIN ANALYZE
SELECT type as vessel_type, count(V.id) as vessels_with_greek_flag
FROM vessels AS V
JOIN positions AS P ON V.id = P.vessel_id
WHERE V.flag = 'Greece'
GROUP BY V.type;
```

Data Output | Messages | Notifications

	QUERY PLAN text
1	Finalize GroupAggregate (cost=189237.82..189245.68 rows=33 width=10) (actual time=1296.612..1300.322 rows=29 loops=1)
2	Group Key: v.type
3	-> Gather Merge (cost=189237.82..189245.02 rows=66 width=10) (actual time=1296.607..1300.305 rows=86 loops=1)
4	Workers Planned: 2
5	Workers Launched: 2
6	-> Sort (cost=188237.29..188237.37 rows=33 width=10) (actual time=1278.012..1278.015 rows=29 loops=3)
7	Sort Key: v.type
8	Sort Method: quicksort Memory: 26kB
9	Worker 0: Sort Method: quicksort Memory: 26kB
10	Worker 1: Sort Method: quicksort Memory: 26kB
11	-> Partial HashAggregate (cost=188236.13..188236.46 rows=33 width=10) (actual time=1277.948..1277.953 rows=29 loops=3)
12	Group Key: v.type
13	Batches: 1 Memory Usage: 24kB
14	Worker 0: Batches: 1 Memory Usage: 24kB
15	Worker 1: Batches: 1 Memory Usage: 24kB
16	-> Hash Join (cost=16.16..180921.27 rows=1462971 width=67) (actual time=0.419..972.178 rows=1824966 loops=3)
17	Hash Cond: ((p.vessel_id)::text = (v.id)::text)
18	-> Parallel Append (cost=0.00..178142.07 rows=2931938 width=65) (actual time=0.121..513.513 rows=2345550 loops=3)
19	-> Parallel Seq Scan on augustweek1 p_1 (cost=0.00..65625.35 rows=1222635 width=65) (actual time=0.201..172.665 rows=978108 loo...
20	-> Parallel Seq Scan on augustweek2 p_2 (cost=0.00..36253.65 rows=666665 width=65) (actual time=0.291..273.385 rows=1599996 loo...
21	-> Parallel Seq Scan on augustweek4 p_4 (cost=0.00..28757.38 rows=529938 width=65) (actual time=0.227..102.065 rows=635926 loop...
22	-> Parallel Seq Scan on augustweek3 p_3 (cost=0.00..27846.00 rows=512700 width=65) (actual time=0.058..203.647 rows=1230480 loo...
23	-> Hash (cost=13.11..13.11 rows=244 width=67) (actual time=0.277..0.277 rows=244 loops=3)
24	Buckets: 1024 Batches: 1 Memory Usage: 32kB
25	-> Seq Scan on vessels v (cost=0.00..13.11 rows=244 width=67) (actual time=0.179..0.235 rows=244 loops=3)
26	Filter: ((flag)::text = 'Greece'::text)
27	Rows Removed by Filter: 245
28	Planning Time: 0.243 ms
29	Execution Time: 1300.369 ms

```
2  EXPLAIN ANALYZE
3  SELECT P.t::date AS given_day, count(lon) AS registered_vessel_pisition
4  FROM positions AS P
5  JOIN vessels AS V on P.vessel_id = V.id
6  JOIN vessel_types AS VT ON  V.type = VT.code
7  WHERE VT.description LIKE 'Passenger,%'
8      AND P.t::date BETWEEN '2019-08-14' AND '2019-08-18'
9  GROUP BY P.t::date;
```

Data Output | Messages | Notifications

	QUERY PLAN text
1	Finalize GroupAggregate (cost=189035.69..189099.23 rows=200 width=12) (actual time=635.813..678.596 rows=5 loops=1)
2	Group Key: ((p.t)::date)
3	-> Gather Merge (cost=189035.69..189094.73 rows=400 width=12) (actual time=635.809..678.580 rows=6 loops=1)
4	Workers Planned: 2
5	Workers Launched: 2
6	-> Partial GroupAggregate (cost=188035.66..188048.54 rows=200 width=12) (actual time=600.709..614.950 rows=2 loops=3)
7	Group Key: ((p.t)::date)
8	-> Sort (cost=188035.66..188039.12 rows=1383 width=12) (actual time=595.722..605.070 rows=120498 loops=3)
9	Sort Key: ((p.t)::date)
10	Sort Method: external merge Disk: 7504kB
11	Worker 0: Sort Method: quicksort Memory: 360kB
12	Worker 1: Sort Method: quicksort Memory: 1586kB
13	-> Hash Join (cost=16.24..187963.52 rows=1383 width=12) (actual time=182.684..575.254 rows=120498 loops=3)
14	Hash Cond: ((p.vessel_id)::text = (v.id)::text)
15	-> Parallel Append (cost=0.00..187875.05 rows=14659 width=81) (actual time=181.801..515.558 rows=392404 loops=3)
16	-> Parallel Seq Scan on augustweek1 p_1 (cost=0.00..77851.69 rows=6113 width=81) (actual time=544.574..544.575 rows=0 loops=1)
17	Filter: (((t)::date >= '2019-08-14'::date) AND ((t)::date <= '2019-08-18'::date))
18	Rows Removed by Filter: 2934323
19	-> Parallel Seq Scan on augustweek2 p_2 (cost=0.00..42920.30 rows=3333 width=81) (actual time=0.551..301.065 rows=72994 loops...
20	Filter: (((t)::date >= '2019-08-14'::date) AND ((t)::date <= '2019-08-18'::date))
21	Rows Removed by Filter: 1527002
22	-> Parallel Seq Scan on augustweek4 p_4 (cost=0.00..34056.77 rows=2650 width=81) (actual time=94.139..94.139 rows=0 loops=3)
23	Filter: (((t)::date >= '2019-08-14'::date) AND ((t)::date <= '2019-08-18'::date))
24	Rows Removed by Filter: 423951
25	-> Parallel Seq Scan on augustweek3 p_3 (cost=0.00..32973.00 rows=2563 width=81) (actual time=0.135..181.170 rows=552109 loo...
26	Filter: (((t)::date >= '2019-08-14'::date) AND ((t)::date <= '2019-08-18'::date))
27	Rows Removed by Filter: 63131
28	-> Hash (cost=15.67..15.67 rows=46 width=65) (actual time=0.823..0.825 rows=64 loops=3)
29	Buckets: 1024 Batches: 1 Memory Usage: 15kB
30	-> Hash Join (cost=2.45..15.67 rows=46 width=65) (actual time=0.692..0.808 rows=64 loops=3)
31	Hash Cond: ((v.type)::text = (vt.code)::text)
32	-> Seq Scan on vessels v (cost=0.00..11.89 rows=489 width=67) (actual time=0.380..0.437 rows=489 loops=3)
33	-> Hash (cost=2.33..2.33 rows=10 width=2) (actual time=0.296..0.297 rows=10 loops=3)
34	Buckets: 1024 Batches: 1 Memory Usage: 9kB
35	-> Seq Scan on vessel_types vt (cost=0.00..2.33 rows=10 width=2) (actual time=0.280..0.286 rows=10 loops=3)
36	Filter: ((description)::text ~~ 'Passenger,%'::text)
37	Rows Removed by Filter: 96
38	Planning Time: 0.549 ms
39	Execution Time: 680.279 ms

```
2  EXPLAIN ANALYZE
3  SELECT V.id as vessel_id, P.speed as stationary, V.type as cargo_vessels
4  FROM positions AS P
5  JOIN vessels AS V on P.vessel_id = V.id
6  WHERE V.type BETWEEN '70' and '79'
7      AND P.speed = 0.0
8      AND P.t >= '2019-08-15 00:00:00' AND P.t <= '2019-08-18 00:00:00'
9  GROUP BY V.id, P.speed;
```

Data Output | Messages | Notifications

	QUERY PLAN text
1	Group (cost=33362.99..33388.01 rows=105 width=71) (actual time=156.877..161.260 rows=30 loops=1)
2	Group Key: v.id
3	-> Gather Merge (cost=33362.99..33387.49 rows=210 width=71) (actual time=156.875..161.246 rows=86 loops=1)
4	Workers Planned: 2
5	Workers Launched: 2
6	-> Sort (cost=32362.96..32363.23 rows=105 width=71) (actual time=137.514..137.517 rows=29 loops=3)
7	Sort Key: v.id
8	Sort Method: quicksort Memory: 27kB
9	Worker 0: Sort Method: quicksort Memory: 27kB
10	Worker 1: Sort Method: quicksort Memory: 27kB
11	-> Partial HashAggregate (cost=32358.39..32359.44 rows=105 width=71) (actual time=137.434..137.440 rows=29 loops=3)
12	Group Key: v.id
13	Batches: 1 Memory Usage: 24kB
14	Worker 0: Batches: 1 Memory Usage: 24kB
15	Worker 1: Batches: 1 Memory Usage: 24kB
16	-> Hash Join (cost=15.65..32248.57 rows=43928 width=71) (actual time=0.767..135.936 rows=7094 loops=3)
17	Hash Cond: ((p.vessel_id)::text = (v.id)::text)
18	-> Parallel Seq Scan on augustweek3 p (cost=0.00..31691.25 rows=204579 width=69) (actual time=0.270..118.009 rows=160205 loops=3)
19	Filter: ((t >= '2019-08-15 00:00:00'::timestamp without time zone) AND (t <= '2019-08-18 00:00:00'::timestamp without time zone) AND (speed =...
20	Rows Removed by Filter: 249955
21	-> Hash (cost=14.34..14.34 rows=105 width=67) (actual time=0.398..0.399 rows=104 loops=3)
22	Buckets: 1024 Batches: 1 Memory Usage: 19kB
23	-> Seq Scan on vessels v (cost=0.00..14.34 rows=105 width=67) (actual time=0.203..0.373 rows=104 loops=3)
24	Filter: (((type)::text >= '70'::text) AND ((type)::text <= '79'::text))
25	Rows Removed by Filter: 385
26	Planning Time: 0.320 ms
27	Execution Time: 161.312 ms

```
2  EXPLAIN ANALYZE
3  SELECT P.vessel_id as completely_stationary_vessels_between_12_and_18_August2019
4  FROM positions AS P
5  JOIN vessels AS V on P.vessel_id = V.id
6  WHERE V.type::integer BETWEEN 70 and 79
7      AND P.t >= '2019-08-12 00:00:00' AND P.t <= '2019-08-19 00:00:00'
8  GROUP BY P.vessel_id
9  HAVING MAX(P.speed) = 0.0;
```

Data Output | Messages | Notifications

	QUERY PLAN text
1	Finalize GroupAggregate (cost=74094.03..74165.42 rows=1 width=65) (actual time=269.659..272.312 rows=1 loops=1)
2	Group Key: p.vessel_id
3	Filter: (max(p.speed) = 0.0)
4	Rows Removed by Filter: 48
5	-> Gather Merge (cost=74094.03..74160.92 rows=400 width=97) (actual time=262.089..272.259 rows=122 loops=1)
6	Workers Planned: 2
7	Workers Launched: 2
8	-> Partial GroupAggregate (cost=73094.00..73114.73 rows=200 width=97) (actual time=230.068..237.140 rows=41 loops=3)
9	Group Key: p.vessel_id
10	-> Sort (cost=73094.00..73100.24 rows=2497 width=69) (actual time=230.004..231.842 rows=29614 loops=3)
11	Sort Key: p.vessel_id
12	Sort Method: quicksort Memory: 4078kB
13	Worker 0: Sort Method: quicksort Memory: 3079kB
14	Worker 1: Sort Method: quicksort Memory: 3528kB
15	-> Hash Join (cost=19.25..72953.10 rows=2497 width=69) (actual time=4.484..200.887 rows=29614 loops=3)
16	Hash Cond: ((p.vessel_id)::text = (v.id)::text)
17	-> Parallel Append (cost=0.00..71317.28 rows=610543 width=69) (actual time=3.993..152.517 rows=488680 loops=3)
18	-> Parallel Bitmap Heap Scan on augustweek2 p_1 (cost=6003.05..37855.06 rows=151001 width=69) (actual time=11.675..88.296 rows...
19	Recheck Cond: ((t >= '2019-08-12 00:00:00'::timestamp without time zone) AND (t <= '2019-08-19 00:00:00'::timestamp without time z...
20	-> Bitmap Index Scan on augustweek2_t_idx (cost=0.00..5912.45 rows=362402 width=0) (actual time=10.987..10.987 rows=361823 l...
21	Index Cond: ((t >= '2019-08-12 00:00:00'::timestamp without time zone) AND (t <= '2019-08-19 00:00:00'::timestamp without time z...
22	-> Parallel Seq Scan on augustweek3 p_2 (cost=0.00..30409.50 rows=459542 width=69) (actual time=0.274..99.962 rows=368073 loops...
23	Filter: ((t >= '2019-08-12 00:00:00'::timestamp without time zone) AND (t <= '2019-08-19 00:00:00'::timestamp without time zone))
24	Rows Removed by Filter: 42087
25	-> Hash (cost=19.23..19.23 rows=2 width=65) (actual time=0.377..0.378 rows=104 loops=3)
26	Buckets: 1024 Batches: 1 Memory Usage: 18kB
27	-> Seq Scan on vessels v (cost=0.00..19.23 rows=2 width=65) (actual time=0.276..0.356 rows=104 loops=3)
28	Filter: (((type)::integer >= 70) AND ((type)::integer <= 79))
29	Rows Removed by Filter: 385
30	Planning Time: 0.316 ms
31	Execution Time: 272.615 ms

Παρατηρούμε ότι στα πρώτα 3 queries χρόνος εκτέλεσης αυξήθηκε ενώ στα τελευταία 3 ο χρόνος μειώθηκε δραματικά. Αυτό συνέβη διότι η χρήση των partitions από την μια καθυστέρησε την αναζήτηση καθώς δεν γίνεται χρήση της στήλης **t** (timestamp) ενώ από την άλλη βοήθησε καθώς ζητάμε συγκεκριμένες χρονικές περιόδους στα queries μας στη βάση.

	No Partitioning	Partitioned table *Positions*	Partition key used in the query
Query 1	890ms	1029ms	no
Query 2	1222ms	1300ms	no
Query 3	609ms	1300ms	no
Query 4	639ms	680ms	yes
Query 5A	522ms	161ms	yes
Query 5B	600ms	272ms	yes

Πηγές

- https://api.vtexplorer.com/docs/ref-aistypes.html
- https://www.postgresql.org/docs/current/datatype-numeric.html
- https://stackoverflow.com/questions/54031813/i-am-trying-to-copy-a-file-butgetting-error-message
- https://stackoverflow.com/questions/23310478/psql-error-could-not-open-fileaddress-csv-for-reading-no-such-file-or-direc
- https://www.postgresql.org/docs/current/sql-keywords-appendix.html
- https://stackoverflow.com/questions/47170516/postgrex-error-error-58p01undefined-file
- https://www.postgresql.org/docs/current/functions-datetime.html#FUNCTIONSDATETIME-TRUNC
- https://stackoverflow.com/questions/14770829/grouping-timestamps-by-day-not-bytime
- https://www.postgresql.org/docs/current/sql-vacuum.html
- How to set postgreSQL shared_buffers to use more RAM
- What is the unit of PostgreSQL shared_buffers
- How to restart PostgreSQL server, Windows 11
- https://www.postgresql.org/docs/9.1/functions-admin.html
- https://stackoverflow.com/questions/18907047/postgres-db-size-command
- https://medium.com/geekculture/indexing-in-postgres-db-4cf502ce1b4e
- https://medium.com/@dmitry.romanoff/partitioning-a-table-by-range-in-thepostgresql-database-423adb0319b7
- https://stackoverflow.com/questions/23876479/will-postgresql-generate-indexautomatically
- https://stackoverflow.com/questions/42898988/is-primary-key-automaticallyindexed-in-postgresql
- https://www.postgresql.org/docs/8.3/indexes-ordering.html
- https://stackoverflow.com/questions/398884/postgresql-hash-index

- https://www.freecodecamp.org/news/postgresql-indexing-strategies/
- https://www.youtube.com/watch?v=HubezKbFL7E&list=PLfkvOiuhoOfVRU4Uoh9u1zHpluAXa3_k
- https://stackoverflow.com/questions/18483859/postgresql-timestamp-index-notused-when-cast-to-date
- https://stackoverflow.com/questions/76566911/postgres-range-partitioning
- https://www.postgresql.org/docs/current/ddl-partitioning.html
- https://stackoverflow.com/questions/57175646/adding-primary-key-for-partitiontable-in-postgresql