ΒΑΣΕΙΣ ΔΕΔΟΜΕΝΩΝ

ΕΡΓΑΣΤΗΡΙΑΚΗ ΑΝΑΦΟΡΑ

Β ΦΑΣΗ ΕΡΓΑΣΤΗΡΙΑΚΗΣ ΑΣΚΗΣΗ

ΚΑΡΑΛΗΣ ΑΣΤΕΡΙΝΟΣ

ΑΜ: 2020030107

ΣΗΜΑΝΤΙΚΈΣ ΠΑΡΑΤΗΡΉΣΕΙΣ - READ ΜΕ!!! :

Η άσκηση έγινε σε virtual machine με περιορισμένα specs, συνεπώς εάν δοκιμάσετε να τρέξετε τα query ενδέχεται να κάνουν καλύτερους χρόνους από αυτούς που κατέγραψα εγώ.

Στις αρχικές σελίδες της αναφοράς βρίσκεται το βασικό της κομμάτι σχετικά με την περιγραφή της υλοποίησης των βημάτων. Τα κυριότερα κομμάτια της αναφοράς βρίσκονται ***εδώ (σελ 2 εως 5)*** *.Το υπόλοιπο κομμάτι αποτελείται από αναλυτικές μετρήσεις όπως αυτές προέκυψαν από το pgAdmin4.*

(Για αυτό το λόγο η αναφορά έχει τόσες πολλές σελίδες).

ΒΉΜΑ 1^{Ο}

Το αρχικό QUERY που υλοποιήθηκε για το αίτημα της β φάσης:

```
SELECT c.name,
  CASE WHEN c.population > 50000 THEN COALESCE(total_students, 0) ELSE 0 END AS total_students
FROM "Cities" c
LEFT JOIN (
    SELECT p.city_id, COUNT(DISTINCT s.amka) AS total_students
    FROM "Person" p
    JOIN "Student" s ON p.amka = s.amka
    JOIN (
        SELECT "StudentAMKA"
        FROM "Joins" js
        JOIN "Program" pgr ON pgr."ProgramID" = js."ProgramID"
        WHERE js."StudentAMKA" IN (
            SELECT amka
            FROM "Student"
            WHERE "entry_date" BETWEEN '2040-09-01' AND '2050-09-30'
        ) AND pgr."Duration" = 5
        GROUP BY "StudentAMKA"
        HAVING COUNT(js."ProgramID") >= 2
    ) j ON s.amka = j."StudentAMKA"
    GROUP BY p.city_id
) counts ON c.id = counts.city_id
ORDER BY total_students DESC;
```

Αλλαγμένο QUERY στο οποίο έγινε αλλαγή της σειράς των join που χρησιμοποιήται πιο μετά για μετρήσεις στο βήμα 2ο και στο βήμα 3ο :

```
SELECT c.name,
  CASE WHEN c.population > 50000 THEN COALESCE(total_students, 0) ELSE 0 END AS total_students
FROM "Cities" c
LEFT JOIN (
    SELECT p.city_id, COUNT(DISTINCT s.amka) AS total_students
    FROM "Person" p
    JOIN (
        SELECT "StudentAMKA"
        FROM "Joins" js
        JOIN "Program" pgr ON pgr."ProgramID" = js."ProgramID"
        WHERE js."StudentAMKA" IN (
            SELECT amka
            FROM "Student"
            WHERE "entry_date" BETWEEN '2040-09-01' AND '2050-09-30'
        ) AND pgr."Duration" = 5
        GROUP BY "StudentAMKA"
        HAVING COUNT(js."ProgramID") >= 2
    ) j ON p.amka = j."StudentAMKA"
    JOIN "Student" s ON p.amka = s.amka
    GROUP BY p.city_id
) counts ON c.id = counts.city_id
ORDER BY total_students DESC;
```

Αλλαγή το join του Student, μετακινήθηκε πιο χαμηλά στο Query.

ΒΗΜΑ 2ο

Αρχικά τρέχουμε το query όπως είναι **χωρίς καμία τροποποίηση** και προκύπτουν οι χρόνοι: **23s, 11s, 10s**

Έχοντας καταλήξει στο παραπάνω query μπορούμε συμπεράνουμε τα εξής σχετικά με τον τύπο ευρετηρίου που μπορούμε να χρησιμοποιήσουμε και σε ποια σημεία:

B-Tree indexes:

Τα b-tree indexes μπορούν να εφαρμοστούν σε περιπτώσεις ισοτήτων και ανισοτήτων σύμφωνα με το documentation της PostgreSQL, συνεπώς καλύπτουν όλες τις περιπτώσεις μέσα στο query.

Τα indexes τα οποία δοκίμασα είναι τα εξής:

CREATE INDEX idx_cities_id ON "Cities" (id); **με χρόνους: 20s, 10s, 10s**

CREATE INDEX idx_cities_population ON "Cities" (population); **με χρόνους: 20s, 10s, 10s**

CREATE INDEX idx_person_amka ON "Person" (amka); **με χρόνους: 20s, 10s, 10s**

CREATE INDEX idx_student_amka ON "Student" (amka); **με χρόνους: 19s, 11s, 10s**

CREATE INDEX idx_student_entry_date ON "Student" (entry_date); ***με χρόνους: 4s, 2s, 2s***

CREATE INDEX idx_joins_studentamka ON "Joins" ("StudentAMKA");**με χρόνους:3s, 3s, 2s**

CREATE INDEX idx_program_programid ON "Program" ("ProgramID");**με χρόνους: 3s, 2s, 2s**

CREATE INDEX idx_program_duration ON "Program" ("Duration");**με χρόνους: 2,7s , 2s, 2s**

Από τους παραπάνω χρόνους συμπεραίνουμε ότι τον καλύτερο χρόνο κάνει το index για το entry_date.

HASH indexes:

Τα hash indexes είναι σχεδιασμένα και προσαρμοσμένα σε περιπτώσεις ισοτήτων μόνο σύμφωνα με το documentation της PostgreSQL συνεπώς καλύπτουν συγκεκριμένες περιπτώσεις μέσα στο query.

Τα indexes τα οποία δοκίμασα είναι τα εξής:

CREATE INDEX idx_person_amka ON "Person" USING hash (amka); **με χρόνους: 25s, 10s, 11s**

CREATE INDEX idx_student_amka ON "Student" USING hash (amka); **με χρόνους:15s, 6s, 5s**

CREATE INDEX idx_program_duration ON "Program" USING hash ("Duration");

με χρόνους: 5s, 5s, 7s

CREATE INDEX idx_program_programid ON "Program" USING hash ("ProgramID");

με χρόνους: 6s, 6s, 8s

Από τους παραπάνω χρόνους συμπεραίνουμε ότι τον καλύτερο χρόνο κάνει το index για το Duration.

Χρήση των βέλτιστων ευρετηρίων από **hash(Duration) και b-tree(entry_date) χωρίς clusterting με χρόνους: 2,5s , 2,4s , 2,5s**

Χρήση των βέλτιστων ευρετηρίων από **hash και b-tree με clusterting με χρόνους: 2,4s , 1,1s , 1,5s**

Χρήση των βέλτιστων ευρετηρίων από **hash και b-tree με clusterting με χρόνους** και με **αλλαγμένη σειρα των join** στα tables**: 2,6s 1,9s 1,4s**

Clusters που χρησιμοποιήθηκαν:

CLUSTER "Student" USING idx_student_entry_date;

CLUSTER "Program" USING idx_program_duration;

ΣΥΜΠΑΊΡΑΣΜΑ ΑΠΟ ΒΉΜΑ 2^{Ο} :

Το **καλύτερο ευρετήριο** βάση των μετρήσεων είναι το index τύπου **B-Tree για το entry_date** του Student, καθώς κάνει την μεγαλύτερη διαφορά.

Καλύτερη απόδοση (μικρότερο execution time) έχουμε όταν χρησιμοποιούμε **hash index στο duration και b-tree index στο entry_date μαζί** με **clustering.**

ΒΉΜΑ 3^{ο}

Για την εισαγωγή μεγάλους όγκου υλοποιήθηκαν 3 διαφορετικές συναρτήσεις:

add_person_entries προσθήκη στο πίνακα Person.

add_program_entries προσθήκη στο πίνακα Program

add_student_joins_entries προσθήκη στο πίνακα Student και Joins

Η τελευταία συνάρτηση κάνει ταυτόχρονα εισαγωγή δεδομένων και στους δύο πίνακες καθώς η ξεχωριστή υλοποίηση της add_joins_entries έκανε πολύ χρόνο (15 mins+)

Τελικά **έγινε εισαγωγή 800.000** entries **στο Person**, **500.000** entries **στο Program**, **500.000 στα Joins και Student** αντίστοιχα.

Αρχικά τρέχουμε το query όπως είναι **χωρίς καμία τροποποίηση** και προκύπτουν οι χρόνοι: **11s, 12s, 10s**

Με χρήση **των B-Tree indexes** στο **Duration και entry_date** (καθώς αυτά έκανα αισθητές διαφορές στις αρχικές μετρήσεις). **Χρόνοι: 7,2s 6,2s 6,1s**

Με χρήση **των HASH indexes** στο **Duration και StudentAMKA** (καθώς αυτά έκανα αισθητές διαφορές στις αρχικές μετρήσεις). **Χρόνοι: 6,6s 6,7s 6,1s**

Χρήση των βέλτιστων ευρετηρίων από **hash(Duration) και b-tree(entry_date) με clusterting με χρόνους**

: 6,1s 6,3s 6,1s

Χρήση των βέλτιστων ευρετηρίων από **hash και b-tree με clusterting με χρόνους** και με **αλλαγμένη σειρα των join** στα tables**: 15s, 12s, 10s.**

ΣΥΜΠΑΊΡΑΣΜΑ ΑΠΟ ΒΉΜΑ 3^{Ο} :

Μετά την εισαγωγή έξτρα δεδομένων παρατηρούμε ότι **οι χρόνοι ανεβαίνουν σε όλες τις μετρήσεις**. Και **οι δύο τύποι ευρετηρίων** κάνουν πολύ **παρόμοιους χρόνους μεταξύ τους**. Χρησιμοποιώντας τον **καλύτερο τρόπο από την πρώτη φάση** παρατηρούμε ότι **κάνει τον καλύτερο χρόνο και σε αυτήν την περίπτωση, ωστόσο είναι πιο αργός σε σχέση με πριν**. Η **αλλαγή των joins δεν βελτιώνει την απόδοση.**

ΒΑΣΕΙΣ ΔΕΔΟΜΕΝΩΝ

ΑΝΑΦΟΡΑ Β ΦΑΣΗΣ

ΚΑΡΑΛΗΣ ΑΣΤΕΡΙΝΟΣ 2020030107

ΜΕΤΡΗΣΕΙΣ

EXPLAIN ANALYZE - ΑΝΑΛΥΤΙΚΑ ΑΠΟΤΕΛΕΣΜΑΤΑ - QUERY PLAN

Query χωρίς χρήση ευρετηρίων ή clustering.

ΔΟΚΙΜΗ 1Η

"Sort (cost=82340.76..82341.87 rows=445 width=30) (actual time=23066.838..23066.890 rows=445 loops=1)"

" Sort Key: (CASE WHEN (c.population > '50000'::numeric) THEN COALESCE(counts.total_students, '0'::bigint) ELSE '0'::bigint END) DESC"

" Sort Method: quicksort Memory: 61kB"

" -> Hash Left Join (cost=82307.44..82321.19 rows=445 width=30) (actual time=23064.801..23066.605 rows=445 loops=1)"

"   Hash Cond: (c.id = counts.city_id)"

"   -> Seq Scan on ""Cities"" c (cost=0.00..11.45 rows=445 width=31) (actual time=0.084..1.637 rows=445 loops=1)"

"   -> Hash (cost=82307.09..82307.09 rows=28 width=12) (actual time=23064.230..23064.240 rows=28 loops=1)"

"   Buckets: 1024 Batches: 1 Memory Usage: 10kB"

"   -> Subquery Scan on counts (cost=82299.79..82307.09 rows=28 width=12) (actual time=23063.707..23064.223 rows=28 loops=1)"

"   -> GroupAggregate (cost=82299.79..82306.81 rows=28 width=12) (actual time=23063.706..23064.216 rows=28 loops=1)"

"   Group Key: p.city_id"

"   -> Sort (cost=82299.79..82302.04 rows=899 width=16) (actual time=23061.880..23061.948 rows=553 loops=1)"

"   Sort Key: p.city_id"

"   Sort Method: quicksort Memory: 50kB"

"   -> Nested Loop (cost=77798.36..82255.69 rows=899 width=16) (actual time=22591.042..23059.974 rows=553 loops=1)"

"   -> Nested Loop (cost=77797.93..81652.51 rows=899 width=24) (actual time=22589.012..22671.649 rows=553 loops=1)"

"   -> GroupAggregate (cost=77797.50..77851.46 rows=899 width=12) (actual time=22588.257..22655.925 rows=553 loops=1)"

"   Group Key: js.""StudentAMKA"""

"   Filter: (count(js.""ProgramID"") >= 2)"

"   Rows Removed by Filter: 58972"

" -> Sort (cost=77797.50..77804.25 rows=2698 width=16) (actual time=22588.204..22624.937 rows=60081 loops=1)"

" Sort Key: js.""StudentAMKA"""

" Sort Method: external merge Disk: 1536kB"

" -> Nested Loop (cost=0.85..77643.75 rows=2698 width=16) (actual time=1846.257..22044.505 rows=60081 loops=1)"

" -> Nested Loop (cost=0.43..46016.31 rows=49142 width=16) (actual time=0.603..1935.725 rows=540089 loops=1)"

" -> Seq Scan on ""Program"" pgr (cost=0.00..2.50 rows=1 width=4) (actual time=0.495..0.544 rows=11 loops=1)"

" Filter: (""Duration"" = 5)"

" Rows Removed by Filter: 29"

" -> Index Only Scan using ""Joins_pkey"" on ""Joins"" js (cost=0.43..45522.39 rows=49142 width=16) (actual time=0.130..165.948 rows=49099 loops=11)"

" Index Cond: (""ProgramID"" = pgr.""ProgramID"")"

" Heap Fetches: 0"

" -> Index Scan using ""Student_pkey"" on ""Student"" (cost=0.43..0.64 rows=1 width=12) (actual time=0.037..0.037 rows=0 loops=540089)"

" Index Cond: ((amka)::text = (js.""StudentAMKA"")::text)"

" Filter: ((entry_date >= '2040-09-01'::date) AND (entry_date <= '2050-09-30'::date))"

" Rows Removed by Filter: 1"

" -> Index Only Scan using ""Student_pkey"" on ""Student"" s (cost=0.43..4.22 rows=1 width=12) (actual time=0.024..0.024 rows=1 loops=553)"

" Index Cond: (amka = (js.""StudentAMKA"")::text)"

" Heap Fetches: 0"

" -> Index Scan using ""Person_pkey"" on ""Person"" p (cost=0.43..0.67 rows=1 width=16) (actual time=0.697..0.697 rows=1 loops=553)"

" Index Cond: ((amka)::text = (s.amka)::text)"

"Planning Time: 22.820 ms"

"Execution Time: 23070.321 ms"

ΔΟΚΙΜΗ 2Η

"Sort (cost=82340.76..82341.87 rows=445 width=30) (actual time=11630.261..11630.290 rows=445 loops=1)"

" Sort Key: (CASE WHEN (c.population > '50000'::numeric) THEN COALESCE(counts.total_students, '0'::bigint) ELSE '0'::bigint END) DESC"

" Sort Method: quicksort Memory: 61kB"

" -> Hash Left Join (cost=82307.44..82321.19 rows=445 width=30) (actual time=11629.795..11630.134 rows=445 loops=1)"

"   Hash Cond: (c.id = counts.city_id)"

"   -> Seq Scan on ""Cities"" c (cost=0.00..11.45 rows=445 width=31) (actual time=0.020..0.121 rows=445 loops=1)"

"   -> Hash (cost=82307.09..82307.09 rows=28 width=12) (actual time=11629.724..11629.729 rows=28 loops=1)"

"   Buckets: 1024 Batches: 1 Memory Usage: 10kB"

"   -> Subquery Scan on counts (cost=82299.79..82307.09 rows=28 width=12) (actual time=11629.316..11629.711 rows=28 loops=1)"

"   -> GroupAggregate (cost=82299.79..82306.81 rows=28 width=12) (actual time=11629.314..11629.703 rows=28 loops=1)"

"   Group Key: p.city_id"

"   -> Sort (cost=82299.79..82302.04 rows=899 width=16) (actual time=11627.394..11627.429 rows=553 loops=1)"

"   Sort Key: p.city_id"

"   Sort Method: quicksort Memory: 50kB"

"   -> Nested Loop (cost=77798.36..82255.69 rows=899 width=16) (actual time=11527.325..11627.031 rows=553 loops=1)"

"   -> Nested Loop (cost=77797.93..81652.51 rows=899 width=24) (actual time=11527.268..11605.099 rows=553 loops=1)"

"   -> GroupAggregate (cost=77797.50..77851.46 rows=899 width=12) (actual time=11527.209..11595.776 rows=553 loops=1)"

"   Group Key: js.""StudentAMKA"""

"   Filter: (count(js.""ProgramID"") >= 2)"

"   Rows Removed by Filter: 58972"

"   -> Sort (cost=77797.50..77804.25 rows=2698 width=16) (actual time=11527.169..11565.164 rows=60081 loops=1)"

"   Sort Key: js.""StudentAMKA"""

"   Sort Method: external merge Disk: 1536kB"

"   -> Nested Loop (cost=0.85..77643.75 rows=2698 width=16) (actual time=1790.281..10920.539 rows=60081 loops=1)"

"   -> Nested Loop (cost=0.43..46016.31 rows=49142 width=16) (actual time=0.513..1616.359 rows=540089 loops=1)"

" -> Seq Scan on ""Program"" pgr (cost=0.00..2.50 rows=1 width=4) (actual time=0.039..0.081 rows=11 loops=1)"

" Filter: (""Duration"" = 5)"

" Rows Removed by Filter: 29"

" -> Index Only Scan using ""Joins_pkey"" on ""Joins"" js (cost=0.43..45522.39 rows=49142 width=16) (actual time=0.073..138.479 rows=49099 loops=11)"

" Index Cond: (""ProgramID"" = pgr.""ProgramID"")"

" Heap Fetches: 0"

" -> Index Scan using ""Student_pkey"" on ""Student"" (cost=0.43..0.64 rows=1 width=12) (actual time=0.017..0.017 rows=0 loops=540089)"

" Index Cond: ((amka)::text = (js.""StudentAMKA"")::text)"

" Filter: ((entry_date >= '2040-09-01'::date) AND (entry_date <= '2050-09-30'::date))"

" Rows Removed by Filter: 1"

" -> Index Only Scan using ""Student_pkey"" on ""Student"" s (cost=0.43..4.22 rows=1 width=12) (actual time=0.015..0.015 rows=1 loops=553)"

" Index Cond: (amka = (js.""StudentAMKA"")::text)"

" Heap Fetches: 0"

" -> Index Scan using ""Person_pkey"" on ""Person"" p (cost=0.43..0.67 rows=1 width=16) (actual time=0.039..0.039 rows=1 loops=553)"

" Index Cond: ((amka)::text = (s.amka)::text)"

"Planning Time: 6.895 ms"

"Execution Time: 11631.707 ms"

ΔΟΚΙΜΗ 3Η

"Sort (cost=82340.76..82341.87 rows=445 width=30) (actual time=10651.056..10651.086 rows=445 loops=1)"

" Sort Key: (CASE WHEN (c.population > '50000'::numeric) THEN COALESCE(counts.total_students, '0'::bigint) ELSE '0'::bigint END) DESC"

" Sort Method: quicksort Memory: 61kB"

" -> Hash Left Join (cost=82307.44..82321.19 rows=445 width=30) (actual time=10650.687..10650.978 rows=445 loops=1)"

"   Hash Cond: (c.id = counts.city_id)"

"   -> Seq Scan on ""Cities"" c (cost=0.00..11.45 rows=445 width=31) (actual time=0.118..0.244 rows=445 loops=1)"

"   -> Hash (cost=82307.09..82307.09 rows=28 width=12) (actual time=10650.551..10650.557 rows=28 loops=1)"

"   Buckets: 1024 Batches: 1 Memory Usage: 10kB"

"   -> Subquery Scan on counts (cost=82299.79..82307.09 rows=28 width=12) (actual time=10650.139..10650.537 rows=28 loops=1)"

"   -> GroupAggregate (cost=82299.79..82306.81 rows=28 width=12) (actual time=10650.138..10650.532 rows=28 loops=1)"

"   Group Key: p.city_id"

"   -> Sort (cost=82299.79..82302.04 rows=899 width=16) (actual time=10648.283..10648.322 rows=553 loops=1)"

"   Sort Key: p.city_id"

"   Sort Method: quicksort Memory: 50kB"

"   -> Nested Loop (cost=77798.36..82255.69 rows=899 width=16) (actual time=10546.561..10646.898 rows=553 loops=1)"

"   -> Nested Loop (cost=77797.93..81652.51 rows=899 width=24) (actual time=10546.502..10626.124 rows=553 loops=1)"

"   -> GroupAggregate (cost=77797.50..77851.46 rows=899 width=12) (actual time=10546.443..10617.156 rows=553 loops=1)"

"   Group Key: js.""StudentAMKA"""

"   Filter: (count(js.""ProgramID"") >= 2)"

"   Rows Removed by Filter: 58972"

"   -> Sort (cost=77797.50..77804.25 rows=2698 width=16) (actual time=10546.391..10584.002 rows=60081 loops=1)"

"   Sort Key: js.""StudentAMKA"""

"   Sort Method: external merge Disk: 1536kB"

"   -> Nested Loop (cost=0.85..77643.75 rows=2698 width=16) (actual time=1050.682..9990.876 rows=60081 loops=1)"

"   -> Nested Loop (cost=0.43..46016.31 rows=49142 width=16) (actual time=0.056..1643.970 rows=540089 loops=1)"

" -> Seq Scan on ""Program"" pgr (cost=0.00..2.50 rows=1 width=4) (actual time=0.026..0.064 rows=11 loops=1)"

" Filter: (""Duration"" = 5)"

" Rows Removed by Filter: 29"

" -> Index Only Scan using ""Joins_pkey"" on ""Joins"" js (cost=0.43..45522.39 rows=49142 width=16) (actual time=0.030..141.254 rows=49099 loops=11)"

" Index Cond: (""ProgramID"" = pgr.""ProgramID"")"

" Heap Fetches: 0"

" -> Index Scan using ""Student_pkey"" on ""Student"" (cost=0.43..0.64 rows=1 width=12) (actual time=0.015..0.015 rows=0 loops=540089)"

" Index Cond: ((amka)::text = (js.""StudentAMKA"")::text)"

" Filter: ((entry_date >= '2040-09-01'::date) AND (entry_date <= '2050-09-30'::date))"

" Rows Removed by Filter: 1"

" -> Index Only Scan using ""Student_pkey"" on ""Student"" s (cost=0.43..4.22 rows=1 width=12) (actual time=0.015..0.015 rows=1 loops=553)"

" Index Cond: (amka = (js.""StudentAMKA"")::text)"

" Heap Fetches: 0"

" -> Index Scan using ""Person_pkey"" on ""Person"" p (cost=0.43..0.67 rows=1 width=16) (actual time=0.037..0.037 rows=1 loops=553)"

" Index Cond: ((amka)::text = (s.amka)::text)"

"Planning Time: 1.300 ms"

"Execution Time: 10652.094 ms"

Query με χρήση ευρετηρίων B-Tree:

(Cities: id)

ΔΟΚΙΜΗ 1η

"Sort (cost=82340.73..82341.84 rows=445 width=30) (actual time=14747.608..14747.636 rows=445 loops=1)"

" Sort Key: (CASE WHEN (c.population > '50000'::numeric) THEN COALESCE(counts.total_students, '0'::bigint) ELSE '0'::bigint END) DESC"

" Sort Method: quicksort Memory: 61kB"

" -> Hash Left Join (cost=82307.41..82321.15 rows=445 width=30) (actual time=14746.993..14747.534 rows=445 loops=1)"

"   Hash Cond: (c.id = counts.city_id)"

"   -> Seq Scan on ""Cities"" c (cost=0.00..11.45 rows=445 width=31) (actual time=0.048..0.370 rows=445 loops=1)"

"   -> Hash (cost=82307.06..82307.06 rows=28 width=12) (actual time=14746.934..14746.939 rows=28 loops=1)"

"   Buckets: 1024 Batches: 1 Memory Usage: 10kB"

"   -> Subquery Scan on counts (cost=82299.76..82307.06 rows=28 width=12) (actual time=14746.556..14746.929 rows=28 loops=1)"

"   -> GroupAggregate (cost=82299.76..82306.78 rows=28 width=12) (actual time=14746.556..14746.924 rows=28 loops=1)"

"   Group Key: p.city_id"

"   -> Sort (cost=82299.76..82302.01 rows=899 width=16) (actual time=14744.834..14744.866 rows=553 loops=1)"

"   Sort Key: p.city_id"

"   Sort Method: quicksort Memory: 50kB"

"   -> Nested Loop (cost=77798.33..82255.65 rows=899 width=16) (actual time=14655.748..14744.589 rows=553 loops=1)"

"   -> Nested Loop (cost=77797.90..81652.48 rows=899 width=24) (actual time=14655.656..14727.059 rows=553 loops=1)"

"   -> GroupAggregate (cost=77797.47..77851.43 rows=899 width=12) (actual time=14655.581..14719.109 rows=553 loops=1)"

"   Group Key: js.""StudentAMKA"""

"   Filter: (count(js.""ProgramID"") >= 2)"

"   Rows Removed by Filter: 58972"

"   -> Sort (cost=77797.47..77804.22 rows=2698 width=16) (actual time=14655.506..14690.185 rows=60081 loops=1)"

"   Sort Key: js.""StudentAMKA"""

"   Sort Method: external merge Disk: 1536kB"

"   -> Nested Loop (cost=0.85..77643.72 rows=2698 width=16) (actual time=1791.154..14099.735 rows=60081 loops=1)"

" -> Nested Loop (cost=0.43..46016.22 rows=49142 width=16) (actual time=0.865..1630.621 rows=540089 loops=1)"

" -> Seq Scan on ""Program"" pgr (cost=0.00..2.50 rows=1 width=4) (actual time=0.511..0.549 rows=11 loops=1)"

" Filter: (""Duration"" = 5)"

" Rows Removed by Filter: 29"

" -> Index Only Scan using ""Joins_pkey"" on ""Joins"" js (cost=0.43..45522.30 rows=49142 width=16) (actual time=0.069..139.616 rows=49099 loops=11)"

" Index Cond: (""ProgramID"" = pgr.""ProgramID"")"

" Heap Fetches: 0"

" -> Index Scan using ""Student_pkey"" on ""Student"" (cost=0.43..0.64 rows=1 width=12) (actual time=0.023..0.023 rows=0 loops=540089)"

" Index Cond: ((amka)::text = (js.""StudentAMKA"")::text)"

" Filter: ((entry_date >= '2040-09-01'::date) AND (entry_date <= '2050-09-30'::date))"

" Rows Removed by Filter: 1"

" -> Index Only Scan using ""Student_pkey"" on ""Student"" s (cost=0.43..4.22 rows=1 width=12) (actual time=0.014..0.014 rows=1 loops=553)"

" Index Cond: (amka = (js.""StudentAMKA"")::text)"

" Heap Fetches: 0"

" -> Index Scan using ""Person_pkey"" on ""Person"" p (cost=0.43..0.67 rows=1 width=16) (actual time=0.031..0.031 rows=1 loops=553)"

" Index Cond: ((amka)::text = (s.amka)::text)"

"Planning Time: 4.610 ms"

"Execution Time: 14748.845 ms"

ΔΟΚΙΜΉ 2Η

"Sort (cost=82340.73..82341.84 rows=445 width=30) (actual time=10643.306..10643.342 rows=445 loops=1)"

" Sort Key: (CASE WHEN (c.population > '50000'::numeric) THEN COALESCE(counts.total_students, '0'::bigint) ELSE '0'::bigint END) DESC"

" Sort Method: quicksort Memory: 61kB"

" -> Hash Left Join (cost=82307.41..82321.15 rows=445 width=30) (actual time=10642.893..10643.236 rows=445 loops=1)"

"   Hash Cond: (c.id = counts.city_id)"

"   -> Seq Scan on ""Cities"" c (cost=0.00..11.45 rows=445 width=31) (actual time=0.016..0.181 rows=445 loops=1)"

"   -> Hash (cost=82307.06..82307.06 rows=28 width=12) (actual time=10642.829..10642.840 rows=28 loops=1)"

"   Buckets: 1024 Batches: 1 Memory Usage: 10kB"

"   -> Subquery Scan on counts (cost=82299.76..82307.06 rows=28 width=12) (actual time=10642.417..10642.825 rows=28 loops=1)"

"   -> GroupAggregate (cost=82299.76..82306.78 rows=28 width=12) (actual time=10642.415..10642.818 rows=28 loops=1)"

"   Group Key: p.city_id"

"   -> Sort (cost=82299.76..82302.01 rows=899 width=16) (actual time=10640.044..10640.095 rows=553 loops=1)"

"   Sort Key: p.city_id"

"   Sort Method: quicksort Memory: 50kB"

"   -> Nested Loop (cost=77798.33..82255.65 rows=899 width=16) (actual time=10546.620..10639.761 rows=553 loops=1)"

"   -> Nested Loop (cost=77797.90..81652.48 rows=899 width=24) (actual time=10546.561..10621.064 rows=553 loops=1)"

"   -> GroupAggregate (cost=77797.47..77851.43 rows=899 width=12) (actual time=10546.509..10610.332 rows=553 loops=1)"

"   Group Key: js.""StudentAMKA"""

"   Filter: (count(js.""ProgramID"") >= 2)"

"   Rows Removed by Filter: 58972"

"   -> Sort (cost=77797.47..77804.22 rows=2698 width=16) (actual time=10546.471..10580.994 rows=60081 loops=1)"

"   Sort Key: js.""StudentAMKA"""

"   Sort Method: external merge Disk: 1536kB"

"   -> Nested Loop (cost=0.85..77643.72 rows=2698 width=16) (actual time=995.354..9914.306 rows=60081 loops=1)"

"   -> Nested Loop (cost=0.43..46016.22 rows=49142 width=16) (actual time=0.034..1615.480 rows=540089 loops=1)"

" -> Seq Scan on ""Program"" pgr (cost=0.00..2.50 rows=1 width=4) (actual time=0.016..0.053 rows=11 loops=1)"

" Filter: (""Duration"" = 5)"

" Rows Removed by Filter: 29"

" -> Index Only Scan using ""Joins_pkey"" on ""Joins"" js (cost=0.43..45522.30 rows=49142 width=16) (actual time=0.032..138.018 rows=49099 loops=11)"

" Index Cond: (""ProgramID"" = pgr.""ProgramID"")"

" Heap Fetches: 0"

" -> Index Scan using ""Student_pkey"" on ""Student"" (cost=0.43..0.64 rows=1 width=12) (actual time=0.015..0.015 rows=0 loops=540089)"

" Index Cond: ((amka)::text = (js.""StudentAMKA"")::text)"

" Filter: ((entry_date >= '2040-09-01'::date) AND (entry_date <= '2050-09-30'::date))"

" Rows Removed by Filter: 1"

" -> Index Only Scan using ""Student_pkey"" on ""Student"" s (cost=0.43..4.22 rows=1 width=12) (actual time=0.016..0.016 rows=1 loops=553)"

" Index Cond: (amka = (js.""StudentAMKA"")::text)"

" Heap Fetches: 0"

" -> Index Scan using ""Person_pkey"" on ""Person"" p (cost=0.43..0.67 rows=1 width=16) (actual time=0.033..0.033 rows=1 loops=553)"

" Index Cond: ((amka)::text = (s.amka)::text)"

"Planning Time: 1.297 ms"

"Execution Time: 10646.256 ms"

ΔΟΚΙΜΗ 3Η

"Sort (cost=82340.73..82341.84 rows=445 width=30) (actual time=10940.234..10940.262 rows=445 loops=1)"

" Sort Key: (CASE WHEN (c.population > '50000'::numeric) THEN COALESCE(counts.total_students, '0'::bigint) ELSE '0'::bigint END) DESC"

" Sort Method: quicksort Memory: 61kB"

" -> Hash Left Join (cost=82307.41..82321.15 rows=445 width=30) (actual time=10939.893..10940.159 rows=445 loops=1)"

"   Hash Cond: (c.id = counts.city_id)"

"   -> Seq Scan on ""Cities"" c (cost=0.00..11.45 rows=445 width=31) (actual time=0.026..0.112 rows=445 loops=1)"

"   -> Hash (cost=82307.06..82307.06 rows=28 width=12) (actual time=10939.855..10939.859 rows=28 loops=1)"

"   Buckets: 1024 Batches: 1 Memory Usage: 10kB"

"   -> Subquery Scan on counts (cost=82299.76..82307.06 rows=28 width=12) (actual time=10939.473..10939.848 rows=28 loops=1)"

"   -> GroupAggregate (cost=82299.76..82306.78 rows=28 width=12) (actual time=10939.472..10939.844 rows=28 loops=1)"

"   Group Key: p.city_id"

"   -> Sort (cost=82299.76..82302.01 rows=899 width=16) (actual time=10937.681..10937.715 rows=553 loops=1)"

"   Sort Key: p.city_id"

"   Sort Method: quicksort Memory: 50kB"

"   -> Nested Loop (cost=77798.33..82255.65 rows=899 width=16) (actual time=10841.146..10937.366 rows=553 loops=1)"

"   -> Nested Loop (cost=77797.90..81652.48 rows=899 width=24) (actual time=10841.079..10915.707 rows=553 loops=1)"

"   -> GroupAggregate (cost=77797.47..77851.43 rows=899 width=12) (actual time=10841.017..10907.264 rows=553 loops=1)"

"   Group Key: js.""StudentAMKA"""

"   Filter: (count(js.""ProgramID"") >= 2)"

"   Rows Removed by Filter: 58972"

"   -> Sort (cost=77797.47..77804.22 rows=2698 width=16) (actual time=10840.977..10876.428 rows=60081 loops=1)"

"   Sort Key: js.""StudentAMKA"""

"   Sort Method: external merge Disk: 1536kB"

"   -> Nested Loop (cost=0.85..77643.72 rows=2698 width=16) (actual time=1138.548..10233.276 rows=60081 loops=1)"

"   -> Nested Loop (cost=0.43..46016.22 rows=49142 width=16) (actual time=0.048..1677.567 rows=540089 loops=1)"

" -> Seq Scan on ""Program"" pgr (cost=0.00..2.50 rows=1 width=4) (actual time=0.022..0.069 rows=11 loops=1)"

" Filter: (""Duration"" = 5)"

" Rows Removed by Filter: 29"

" -> Index Only Scan using ""Joins_pkey"" on ""Joins"" js (cost=0.43..45522.30 rows=49142 width=16) (actual time=0.037..142.643 rows=49099 loops=11)"

" Index Cond: (""ProgramID"" = pgr.""ProgramID"")"

" Heap Fetches: 0"

" -> Index Scan using ""Student_pkey"" on ""Student"" (cost=0.43..0.64 rows=1 width=12) (actual time=0.015..0.015 rows=0 loops=540089)"

" Index Cond: ((amka)::text = (js.""StudentAMKA"")::text)"

" Filter: ((entry_date >= '2040-09-01'::date) AND (entry_date <= '2050-09-30'::date))"

" Rows Removed by Filter: 1"

" -> Index Only Scan using ""Student_pkey"" on ""Student"" s (cost=0.43..4.22 rows=1 width=12) (actual time=0.014..0.014 rows=1 loops=553)"

" Index Cond: (amka = (js.""StudentAMKA"")::text)"

" Heap Fetches: 0"

" -> Index Scan using ""Person_pkey"" on ""Person"" p (cost=0.43..0.67 rows=1 width=16) (actual time=0.038..0.038 rows=1 loops=553)"

" Index Cond: ((amka)::text = (s.amka)::text)"

"Planning Time: 6.818 ms"

"Execution Time: 10941.378 ms"

(Cities: id, population)

ΔΟΚΙΜΉΗ 1Η

"Sort (cost=82340.73..82341.84 rows=445 width=30) (actual time=10940.234..10940.262 rows=445 loops=1)"

" Sort Key: (CASE WHEN (c.population > '50000'::numeric) THEN COALESCE(counts.total_students, '0'::bigint) ELSE '0'::bigint END) DESC"

" Sort Method: quicksort Memory: 61kB"

" -> Hash Left Join (cost=82307.41..82321.15 rows=445 width=30) (actual time=10939.893..10940.159 rows=445 loops=1)"

"   Hash Cond: (c.id = counts.city_id)"

"   -> Seq Scan on ""Cities"" c (cost=0.00..11.45 rows=445 width=31) (actual time=0.026..0.112 rows=445 loops=1)"

"   -> Hash (cost=82307.06..82307.06 rows=28 width=12) (actual time=10939.855..10939.859 rows=28 loops=1)"

"   Buckets: 1024 Batches: 1 Memory Usage: 10kB"

"   -> Subquery Scan on counts (cost=82299.76..82307.06 rows=28 width=12) (actual time=10939.473..10939.848 rows=28 loops=1)"

"   -> GroupAggregate (cost=82299.76..82306.78 rows=28 width=12) (actual time=10939.472..10939.844 rows=28 loops=1)"

"   Group Key: p.city_id"

"   -> Sort (cost=82299.76..82302.01 rows=899 width=16) (actual time=10937.681..10937.715 rows=553 loops=1)"

"   Sort Key: p.city_id"

"   Sort Method: quicksort Memory: 50kB"

"   -> Nested Loop (cost=77798.33..82255.65 rows=899 width=16) (actual time=10841.146..10937.366 rows=553 loops=1)"

"   -> Nested Loop (cost=77797.90..81652.48 rows=899 width=24) (actual time=10841.079..10915.707 rows=553 loops=1)"

"   -> GroupAggregate (cost=77797.47..77851.43 rows=899 width=12) (actual time=10841.017..10907.264 rows=553 loops=1)"

"   Group Key: js.""StudentAMKA"""

"   Filter: (count(js.""ProgramID"") >= 2)"

"   Rows Removed by Filter: 58972"

"   -> Sort (cost=77797.47..77804.22 rows=2698 width=16) (actual time=10840.977..10876.428 rows=60081 loops=1)"

"   Sort Key: js.""StudentAMKA"""

"   Sort Method: external merge Disk: 1536kB"

"   -> Nested Loop (cost=0.85..77643.72 rows=2698 width=16) (actual time=1138.548..10233.276 rows=60081 loops=1)"

"   -> Nested Loop (cost=0.43..46016.22 rows=49142 width=16) (actual time=0.048..1677.567 rows=540089 loops=1)"

" -> Seq Scan on ""Program"" pgr (cost=0.00..2.50 rows=1 width=4) (actual time=0.022..0.069 rows=11 loops=1)"

" Filter: (""Duration"" = 5)"

" Rows Removed by Filter: 29"

" -> Index Only Scan using ""Joins_pkey"" on ""Joins"" js (cost=0.43..45522.30 rows=49142 width=16) (actual time=0.037..142.643 rows=49099 loops=11)"

" Index Cond: (""ProgramID"" = pgr.""ProgramID"")"

" Heap Fetches: 0"

" -> Index Scan using ""Student_pkey"" on ""Student"" (cost=0.43..0.64 rows=1 width=12) (actual time=0.015..0.015 rows=0 loops=540089)"

" Index Cond: ((amka)::text = (js.""StudentAMKA"")::text)"

" Filter: ((entry_date >= '2040-09-01'::date) AND (entry_date <= '2050-09-30'::date))"

" Rows Removed by Filter: 1"

" -> Index Only Scan using ""Student_pkey"" on ""Student"" s (cost=0.43..4.22 rows=1 width=12) (actual time=0.014..0.014 rows=1 loops=553)"

" Index Cond: (amka = (js.""StudentAMKA"")::text)"

" Heap Fetches: 0"

" -> Index Scan using ""Person_pkey"" on ""Person"" p (cost=0.43..0.67 rows=1 width=16) (actual time=0.038..0.038 rows=1 loops=553)"

" Index Cond: ((amka)::text = (s.amka)::text)"

"Planning Time: 6.818 ms"

"Execution Time: 10941.378 ms"

ΔΟΚΙΜΗ 2Η

"Sort (cost=82340.73..82341.84 rows=445 width=30) (actual time=11351.184..11351.214 rows=445 loops=1)"

" Sort Key: (CASE WHEN (c.population > '50000'::numeric) THEN COALESCE(counts.total_students, '0'::bigint) ELSE '0'::bigint END) DESC"

" Sort Method: quicksort Memory: 61kB"

" -> Hash Left Join (cost=82307.41..82321.15 rows=445 width=30) (actual time=11350.825..11351.108 rows=445 loops=1)"

"   Hash Cond: (c.id = counts.city_id)"

"   -> Seq Scan on ""Cities"" c (cost=0.00..11.45 rows=445 width=31) (actual time=0.022..0.143 rows=445 loops=1)"

"   -> Hash (cost=82307.06..82307.06 rows=28 width=12) (actual time=11350.789..11350.794 rows=28 loops=1)"

"   Buckets: 1024 Batches: 1 Memory Usage: 10kB"

"   -> Subquery Scan on counts (cost=82299.76..82307.06 rows=28 width=12) (actual time=11350.383..11350.772 rows=28 loops=1)"

"   -> GroupAggregate (cost=82299.76..82306.78 rows=28 width=12) (actual time=11350.382..11350.766 rows=28 loops=1)"

"   Group Key: p.city_id"

"   -> Sort (cost=82299.76..82302.01 rows=899 width=16) (actual time=11348.484..11348.524 rows=553 loops=1)"

"   Sort Key: p.city_id"

"   Sort Method: quicksort Memory: 50kB"

"   -> Nested Loop (cost=77798.33..82255.65 rows=899 width=16) (actual time=11244.714..11348.166 rows=553 loops=1)"

"   -> Nested Loop (cost=77797.90..81652.48 rows=899 width=24) (actual time=11244.487..11327.470 rows=553 loops=1)"

"   -> GroupAggregate (cost=77797.47..77851.43 rows=899 width=12) (actual time=11244.434..11311.436 rows=553 loops=1)"

"   Group Key: js.""StudentAMKA"""

"   Filter: (count(js.""ProgramID"") >= 2)"

"   Rows Removed by Filter: 58972"

"   -> Sort (cost=77797.47..77804.22 rows=2698 width=16) (actual time=11244.395..11281.374 rows=60081 loops=1)"

"   Sort Key: js.""StudentAMKA"""

"   Sort Method: external merge Disk: 1536kB"

"   -> Nested Loop (cost=0.85..77643.72 rows=2698 width=16) (actual time=1056.146..10662.610 rows=60081 loops=1)"

"   -> Nested Loop (cost=0.43..46016.22 rows=49142 width=16) (actual time=0.037..1859.915 rows=540089 loops=1)"

" -> Seq Scan on ""Program"" pgr (cost=0.00..2.50 rows=1 width=4) (actual time=0.016..0.068 rows=11 loops=1)"

" Filter: (""Duration"" = 5)"

" Rows Removed by Filter: 29"

" -> Index Only Scan using ""Joins_pkey"" on ""Joins"" js (cost=0.43..45522.30 rows=49142 width=16) (actual time=0.083..160.067 rows=49099 loops=11)"

" Index Cond: (""ProgramID"" = pgr.""ProgramID"")"

" Heap Fetches: 0"

" -> Index Scan using ""Student_pkey"" on ""Student"" (cost=0.43..0.64 rows=1 width=12) (actual time=0.016..0.016 rows=0 loops=540089)"

" Index Cond: ((amka)::text = (js.""StudentAMKA"")::text)"

" Filter: ((entry_date >= '2040-09-01'::date) AND (entry_date <= '2050-09-30'::date))"

" Rows Removed by Filter: 1"

" -> Index Only Scan using ""Student_pkey"" on ""Student"" s (cost=0.43..4.22 rows=1 width=12) (actual time=0.028..0.028 rows=1 loops=553)"

" Index Cond: (amka = (js.""StudentAMKA"")::text)"

" Heap Fetches: 0"

" -> Index Scan using ""Person_pkey"" on ""Person"" p (cost=0.43..0.67 rows=1 width=16) (actual time=0.036..0.036 rows=1 loops=553)"

" Index Cond: ((amka)::text = (s.amka)::text)"

"Planning Time: 0.926 ms"

"Execution Time: 11352.334 ms"

ΔΟΚΙΜΗ 3Η

"Sort (cost=82340.73..82341.84 rows=445 width=30) (actual time=10854.449..10854.476 rows=445 loops=1)"

" Sort Key: (CASE WHEN (c.population > '50000'::numeric) THEN COALESCE(counts.total_students, '0'::bigint) ELSE '0'::bigint END) DESC"

" Sort Method: quicksort Memory: 61kB"

" -> Hash Left Join (cost=82307.41..82321.15 rows=445 width=30) (actual time=10854.140..10854.374 rows=445 loops=1)"

"   Hash Cond: (c.id = counts.city_id)"

"   -> Seq Scan on ""Cities"" c (cost=0.00..11.45 rows=445 width=31) (actual time=0.020..0.098 rows=445 loops=1)"

"   -> Hash (cost=82307.06..82307.06 rows=28 width=12) (actual time=10854.105..10854.109 rows=28 loops=1)"

"   Buckets: 1024 Batches: 1 Memory Usage: 10kB"

"   -> Subquery Scan on counts (cost=82299.76..82307.06 rows=28 width=12) (actual time=10853.722..10854.098 rows=28 loops=1)"

"   -> GroupAggregate (cost=82299.76..82306.78 rows=28 width=12) (actual time=10853.721..10854.093 rows=28 loops=1)"

"   Group Key: p.city_id"

"   -> Sort (cost=82299.76..82302.01 rows=899 width=16) (actual time=10851.978..10852.011 rows=553 loops=1)"

"   Sort Key: p.city_id"

"   Sort Method: quicksort Memory: 50kB"

"   -> Nested Loop (cost=77798.33..82255.65 rows=899 width=16) (actual time=10754.342..10851.682 rows=553 loops=1)"

"   -> Nested Loop (cost=77797.90..81652.48 rows=899 width=24) (actual time=10754.282..10832.027 rows=553 loops=1)"

"   -> GroupAggregate (cost=77797.47..77851.43 rows=899 width=12) (actual time=10754.226..10823.755 rows=553 loops=1)"

"   Group Key: js.""StudentAMKA"""

"   Filter: (count(js.""ProgramID"") >= 2)"

"   Rows Removed by Filter: 58972"

"   -> Sort (cost=77797.47..77804.22 rows=2698 width=16) (actual time=10754.187..10790.367 rows=60081 loops=1)"

"   Sort Key: js.""StudentAMKA"""

"   Sort Method: external merge Disk: 1536kB"

"   -> Nested Loop (cost=0.85..77643.72 rows=2698 width=16) (actual time=1012.405..10215.723 rows=60081 loops=1)"

"   -> Nested Loop (cost=0.43..46016.22 rows=49142 width=16) (actual time=0.037..1655.531 rows=540089 loops=1)"

" -> Seq Scan on ""Program"" pgr (cost=0.00..2.50 rows=1 width=4) (actual time=0.017..0.052 rows=11 loops=1)"

" Filter: (""Duration"" = 5)"

" Rows Removed by Filter: 29"

" -> Index Only Scan using ""Joins_pkey"" on ""Joins"" js (cost=0.43..45522.30 rows=49142 width=16) (actual time=0.033..141.184 rows=49099 loops=11)"

" Index Cond: (""ProgramID"" = pgr.""ProgramID"")"

" Heap Fetches: 0"

" -> Index Scan using ""Student_pkey"" on ""Student"" (cost=0.43..0.64 rows=1 width=12) (actual time=0.015..0.015 rows=0 loops=540089)"

" Index Cond: ((amka)::text = (js.""StudentAMKA"")::text)"

" Filter: ((entry_date >= '2040-09-01'::date) AND (entry_date <= '2050-09-30'::date))"

" Rows Removed by Filter: 1"

" -> Index Only Scan using ""Student_pkey"" on ""Student"" s (cost=0.43..4.22 rows=1 width=12) (actual time=0.014..0.014 rows=1 loops=553)"

" Index Cond: (amka = (js.""StudentAMKA"")::text)"

" Heap Fetches: 0"

" -> Index Scan using ""Person_pkey"" on ""Person"" p (cost=0.43..0.67 rows=1 width=16) (actual time=0.035..0.035 rows=1 loops=553)"

" Index Cond: ((amka)::text = (s.amka)::text)"

"Planning Time: 1.225 ms"

"Execution Time: 10855.362 ms"

(Cities: id, population | Person: amka)

ΔΟΚΙΜΉΗ 1Η

"Sort (cost=82340.73..82341.84 rows=445 width=30) (actual time=23848.104..23848.136 rows=445 loops=1)"

" Sort Key: (CASE WHEN (c.population > '50000'::numeric) THEN COALESCE(counts.total_students, '0'::bigint) ELSE '0'::bigint END) DESC"

" Sort Method: quicksort Memory: 61kB"

" -> Hash Left Join (cost=82307.41..82321.15 rows=445 width=30) (actual time=23847.751..23848.036 rows=445 loops=1)"

"   Hash Cond: (c.id = counts.city_id)"

"   -> Seq Scan on ""Cities"" c (cost=0.00..11.45 rows=445 width=31) (actual time=0.025..0.147 rows=445 loops=1)"

"   -> Hash (cost=82307.06..82307.06 rows=28 width=12) (actual time=23847.717..23847.722 rows=28 loops=1)"

"   Buckets: 1024 Batches: 1 Memory Usage: 10kB"

"   -> Subquery Scan on counts (cost=82299.76..82307.06 rows=28 width=12) (actual time=23847.322..23847.710 rows=28 loops=1)"

"   -> GroupAggregate (cost=82299.76..82306.78 rows=28 width=12) (actual time=23847.321..23847.704 rows=28 loops=1)"

"   Group Key: p.city_id"

"   -> Sort (cost=82299.76..82302.01 rows=899 width=16) (actual time=23845.548..23845.588 rows=553 loops=1)"

"   Sort Key: p.city_id"

"   Sort Method: quicksort Memory: 50kB"

"   -> Nested Loop (cost=77798.33..82255.65 rows=899 width=16) (actual time=23552.574..23844.190 rows=553 loops=1)"

"   -> Nested Loop (cost=77797.90..81652.48 rows=899 width=24) (actual time=23551.575..23632.970 rows=553 loops=1)"

"   -> GroupAggregate (cost=77797.47..77851.43 rows=899 width=12) (actual time=23551.457..23620.441 rows=553 loops=1)"

"   Group Key: js.""StudentAMKA"""

"   Filter: (count(js.""ProgramID"") >= 2)"

"   Rows Removed by Filter: 58972"

"   -> Sort (cost=77797.47..77804.22 rows=2698 width=16) (actual time=23551.394..23590.371 rows=60081 loops=1)"

"   Sort Key: js.""StudentAMKA"""

"   Sort Method: external merge Disk: 1536kB"

"   -> Nested Loop (cost=0.85..77643.72 rows=2698 width=16) (actual time=1896.136..23014.000 rows=60081 loops=1)"

"   -> Nested Loop (cost=0.43..46016.22 rows=49142 width=16) (actual time=0.045..1914.850 rows=540089 loops=1)"

" -> Seq Scan on ""Program"" pgr (cost=0.00..2.50 rows=1 width=4) (actual time=0.023..0.055 rows=11 loops=1)"

" Filter: (""Duration"" = 5)"

" Rows Removed by Filter: 29"

" -> Index Only Scan using ""Joins_pkey"" on ""Joins"" js (cost=0.43..45522.30 rows=49142 width=16) (actual time=0.029..163.939 rows=49099 loops=11)"

" Index Cond: (""ProgramID"" = pgr.""ProgramID"")"

" Heap Fetches: 0"

" -> Index Scan using ""Student_pkey"" on ""Student"" (cost=0.43..0.64 rows=1 width=12) (actual time=0.038..0.038 rows=0 loops=540089)"

" Index Cond: ((amka)::text = (js.""StudentAMKA"")::text)"

" Filter: ((entry_date >= '2040-09-01'::date) AND (entry_date <= '2050-09-30'::date))"

" Rows Removed by Filter: 1"

" -> Index Only Scan using ""Student_pkey"" on ""Student"" s (cost=0.43..4.22 rows=1 width=12) (actual time=0.019..0.019 rows=1 loops=553)"

" Index Cond: (amka = (js.""StudentAMKA"")::text)"

" Heap Fetches: 0"

" -> Index Scan using idx_person_amka on ""Person"" p (cost=0.43..0.67 rows=1 width=16) (actual time=0.378..0.378 rows=1 loops=553)"

" Index Cond: ((amka)::text = (s.amka)::text)"

"Planning Time: 10.486 ms"

"Execution Time: 23849.019 ms"

ΔΟΚΙΜΗ 2Η

"Sort (cost=82340.73..82341.84 rows=445 width=30) (actual time=11003.929..11003.957 rows=445 loops=1)"

" Sort Key: (CASE WHEN (c.population > '50000'::numeric) THEN COALESCE(counts.total_students, '0'::bigint) ELSE '0'::bigint END) DESC"

" Sort Method: quicksort Memory: 61kB"

" -> Hash Left Join (cost=82307.41..82321.15 rows=445 width=30) (actual time=11003.615..11003.859 rows=445 loops=1)"

"   Hash Cond: (c.id = counts.city_id)"

"   -> Seq Scan on ""Cities"" c (cost=0.00..11.45 rows=445 width=31) (actual time=0.019..0.105 rows=445 loops=1)"

"   -> Hash (cost=82307.06..82307.06 rows=28 width=12) (actual time=11003.587..11003.591 rows=28 loops=1)"

"   Buckets: 1024 Batches: 1 Memory Usage: 10kB"

"   -> Subquery Scan on counts (cost=82299.76..82307.06 rows=28 width=12) (actual time=11003.202..11003.580 rows=28 loops=1)"

"   -> GroupAggregate (cost=82299.76..82306.78 rows=28 width=12) (actual time=11003.202..11003.575 rows=28 loops=1)"

"   Group Key: p.city_id"

"   -> Sort (cost=82299.76..82302.01 rows=899 width=16) (actual time=11001.319..11001.353 rows=553 loops=1)"

"   Sort Key: p.city_id"

"   Sort Method: quicksort Memory: 50kB"

"   -> Nested Loop (cost=77798.33..82255.65 rows=899 width=16) (actual time=10898.163..11001.038 rows=553 loops=1)"

"   -> Nested Loop (cost=77797.90..81652.48 rows=899 width=24) (actual time=10898.092..10979.010 rows=553 loops=1)"

"   -> GroupAggregate (cost=77797.47..77851.43 rows=899 width=12) (actual time=10898.032..10970.032 rows=553 loops=1)"

"   Group Key: js.""StudentAMKA"""

"   Filter: (count(js.""ProgramID"") >= 2)"

"   Rows Removed by Filter: 58972"

"   -> Sort (cost=77797.47..77804.22 rows=2698 width=16) (actual time=10897.991..10935.921 rows=60081 loops=1)"

"   Sort Key: js.""StudentAMKA"""

"   Sort Method: external merge Disk: 1536kB"

"   -> Nested Loop (cost=0.85..77643.72 rows=2698 width=16) (actual time=890.120..10341.386 rows=60081 loops=1)"

"   -> Nested Loop (cost=0.43..46016.22 rows=49142 width=16) (actual time=0.039..1657.363 rows=540089 loops=1)"

" -> Seq Scan on ""Program"" pgr (cost=0.00..2.50 rows=1 width=4) (actual time=0.017..0.054 rows=11 loops=1)"

" Filter: (""Duration"" = 5)"

" Rows Removed by Filter: 29"

" -> Index Only Scan using ""Joins_pkey"" on ""Joins"" js (cost=0.43..45522.30 rows=49142 width=16) (actual time=0.040..142.017 rows=49099 loops=11)"

" Index Cond: (""ProgramID"" = pgr.""ProgramID"")"

" Heap Fetches: 0"

" -> Index Scan using ""Student_pkey"" on ""Student"" (cost=0.43..0.64 rows=1 width=12) (actual time=0.016..0.016 rows=0 loops=540089)"

" Index Cond: ((amka)::text = (js.""StudentAMKA"")::text)"

" Filter: ((entry_date >= '2040-09-01'::date) AND (entry_date <= '2050-09-30'::date))"

" Rows Removed by Filter: 1"

" -> Index Only Scan using ""Student_pkey"" on ""Student"" s (cost=0.43..4.22 rows=1 width=12) (actual time=0.015..0.015 rows=1 loops=553)"

" Index Cond: (amka = (js.""StudentAMKA"")::text)"

" Heap Fetches: 0"

" -> Index Scan using idx_person_amka on ""Person"" p (cost=0.43..0.67 rows=1 width=16) (actual time=0.039..0.039 rows=1 loops=553)"

" Index Cond: ((amka)::text = (s.amka)::text)"

"Planning Time: 1.043 ms"

"Execution Time: 11005.089 ms"

ΔΟΚΙΜΗ 3Η

"Sort (cost=82340.73..82341.84 rows=445 width=30) (actual time=11208.270..11208.313 rows=445 loops=1)"

" Sort Key: (CASE WHEN (c.population > '50000'::numeric) THEN COALESCE(counts.total_students, '0'::bigint) ELSE '0'::bigint END) DESC"

" Sort Method: quicksort Memory: 61kB"

" -> Hash Left Join (cost=82307.41..82321.15 rows=445 width=30) (actual time=11207.792..11208.168 rows=445 loops=1)"

"   Hash Cond: (c.id = counts.city_id)"

"   -> Seq Scan on ""Cities"" c (cost=0.00..11.45 rows=445 width=31) (actual time=0.019..0.197 rows=445 loops=1)"

"   -> Hash (cost=82307.06..82307.06 rows=28 width=12) (actual time=11207.753..11207.768 rows=28 loops=1)"

"   Buckets: 1024 Batches: 1 Memory Usage: 10kB"

"   -> Subquery Scan on counts (cost=82299.76..82307.06 rows=28 width=12) (actual time=11205.129..11207.747 rows=28 loops=1)"

"   -> GroupAggregate (cost=82299.76..82306.78 rows=28 width=12) (actual time=11205.128..11207.737 rows=28 loops=1)"

"   Group Key: p.city_id"

"   -> Sort (cost=82299.76..82302.01 rows=899 width=16) (actual time=11203.324..11203.381 rows=553 loops=1)"

"   Sort Key: p.city_id"

"   Sort Method: quicksort Memory: 50kB"

"   -> Nested Loop (cost=77798.33..82255.65 rows=899 width=16) (actual time=11107.972..11203.046 rows=553 loops=1)"

"   -> Nested Loop (cost=77797.90..81652.48 rows=899 width=24) (actual time=11107.910..11181.305 rows=553 loops=1)"

"   -> GroupAggregate (cost=77797.47..77851.43 rows=899 width=12) (actual time=11107.852..11172.693 rows=553 loops=1)"

"   Group Key: js.""StudentAMKA"""

"   Filter: (count(js.""ProgramID"") >= 2)"

"   Rows Removed by Filter: 58972"

"   -> Sort (cost=77797.47..77804.22 rows=2698 width=16) (actual time=11107.814..11143.868 rows=60081 loops=1)"

"   Sort Key: js.""StudentAMKA"""

"   Sort Method: external merge Disk: 1536kB"

"   -> Nested Loop (cost=0.85..77643.72 rows=2698 width=16) (actual time=909.806..10464.707 rows=60081 loops=1)"

"   -> Nested Loop (cost=0.43..46016.22 rows=49142 width=16) (actual time=0.042..1742.547 rows=540089 loops=1)"

" -> Seq Scan on ""Program"" pgr (cost=0.00..2.50 rows=1 width=4) (actual time=0.018..0.061 rows=11 loops=1)"

" Filter: (""Duration"" = 5)"

" Rows Removed by Filter: 29"

" -> Index Only Scan using ""Joins_pkey"" on ""Joins"" js (cost=0.43..45522.30 rows=49142 width=16) (actual time=0.043..150.191 rows=49099 loops=11)"

" Index Cond: (""ProgramID"" = pgr.""ProgramID"")"

" Heap Fetches: 0"

" -> Index Scan using ""Student_pkey"" on ""Student"" (cost=0.43..0.64 rows=1 width=12) (actual time=0.016..0.016 rows=0 loops=540089)"

" Index Cond: ((amka)::text = (js.""StudentAMKA"")::text)"

" Filter: ((entry_date >= '2040-09-01'::date) AND (entry_date <= '2050-09-30'::date))"

" Rows Removed by Filter: 1"

" -> Index Only Scan using ""Student_pkey"" on ""Student"" s (cost=0.43..4.22 rows=1 width=12) (actual time=0.014..0.014 rows=1 loops=553)"

" Index Cond: (amka = (js.""StudentAMKA"")::text)"

" Heap Fetches: 0"

" -> Index Scan using idx_person_amka on ""Person"" p (cost=0.43..0.67 rows=1 width=16) (actual time=0.038..0.038 rows=1 loops=553)"

" Index Cond: ((amka)::text = (s.amka)::text)"

"Planning Time: 6.095 ms"

"Execution Time: 11210.310 ms"

(Cities: id, population | Person: amka | Student: amka)

ΔΟΚΙΜΉΗ 1Η

"Sort (cost=82340.73..82341.84 rows=445 width=30) (actual time=23579.251..23579.275 rows=445 loops=1)"

" Sort Key: (CASE WHEN (c.population > '50000'::numeric) THEN COALESCE(counts.total_students, '0'::bigint) ELSE '0'::bigint END) DESC"

" Sort Method: quicksort Memory: 61kB"

" -> Hash Left Join (cost=82307.41..82321.15 rows=445 width=30) (actual time=23578.937..23579.174 rows=445 loops=1)"

"   Hash Cond: (c.id = counts.city_id)"

"   -> Seq Scan on ""Cities"" c (cost=0.00..11.45 rows=445 width=31) (actual time=0.020..0.098 rows=445 loops=1)"

"   -> Hash (cost=82307.06..82307.06 rows=28 width=12) (actual time=23578.907..23578.910 rows=28 loops=1)"

"   Buckets: 1024 Batches: 1 Memory Usage: 10kB"

"   -> Subquery Scan on counts (cost=82299.76..82307.06 rows=28 width=12) (actual time=23578.524..23578.901 rows=28 loops=1)"

"   -> GroupAggregate (cost=82299.76..82306.78 rows=28 width=12) (actual time=23578.524..23578.896 rows=28 loops=1)"

"   Group Key: p.city_id"

"   -> Sort (cost=82299.76..82302.01 rows=899 width=16) (actual time=23576.782..23576.814 rows=553 loops=1)"

"   Sort Key: p.city_id"

"   Sort Method: quicksort Memory: 50kB"

"   -> Nested Loop (cost=77798.33..82255.65 rows=899 width=16) (actual time=23086.371..23574.490 rows=553 loops=1)"

"   -> Nested Loop (cost=77797.90..81652.48 rows=899 width=24) (actual time=23081.996..23166.981 rows=553 loops=1)"

"   -> GroupAggregate (cost=77797.47..77851.43 rows=899 width=12) (actual time=23080.780..23150.925 rows=553 loops=1)"

"   Group Key: js.""StudentAMKA"""

"   Filter: (count(js.""ProgramID"") >= 2)"

"   Rows Removed by Filter: 58972"

"   -> Sort (cost=77797.47..77804.22 rows=2698 width=16) (actual time=23080.742..23120.026 rows=60081 loops=1)"

"   Sort Key: js.""StudentAMKA"""

"   Sort Method: external merge Disk: 1536kB"

"   -> Nested Loop (cost=0.85..77643.72 rows=2698 width=16) (actual time=2022.823..22562.930 rows=60081 loops=1)"

"   -> Nested Loop (cost=0.43..46016.22 rows=49142 width=16) (actual time=0.042..1894.037 rows=540089 loops=1)"

" -> Seq Scan on ""Program"" pgr (cost=0.00..2.50 rows=1 width=4) (actual time=0.019..0.056 rows=11 loops=1)"

" Filter: (""Duration"" = 5)"

" Rows Removed by Filter: 29"

" -> Index Only Scan using ""Joins_pkey"" on ""Joins"" js (cost=0.43..45522.30 rows=49142 width=16) (actual time=0.042..162.202 rows=49099 loops=11)"

" Index Cond: (""ProgramID"" = pgr.""ProgramID"")"

" Heap Fetches: 0"

" -> Index Scan using idx_student_amka on ""Student"" (cost=0.43..0.64 rows=1 width=12) (actual time=0.038..0.038 rows=0 loops=540089)"

" Index Cond: ((amka)::text = (js.""StudentAMKA"")::text)"

" Filter: ((entry_date >= '2040-09-01'::date) AND (entry_date <= '2050-09-30'::date))"

" Rows Removed by Filter: 1"

" -> Index Only Scan using idx_student_amka on ""Student"" s (cost=0.43..4.22 rows=1 width=12) (actual time=0.024..0.024 rows=1 loops=553)"

" Index Cond: (amka = (js.""StudentAMKA"")::text)"

" Heap Fetches: 0"

" -> Index Scan using idx_person_amka on ""Person"" p (cost=0.43..0.67 rows=1 width=16) (actual time=0.731..0.731 rows=1 loops=553)"

" Index Cond: ((amka)::text = (s.amka)::text)"

"Planning Time: 6.854 ms"

"Execution Time: 23581.985 ms"

ΔΟΚΙΜΉ 2Η

"Sort (cost=82340.73..82341.84 rows=445 width=30) (actual time=11891.814..11891.898 rows=445 loops=1)"

" Sort Key: (CASE WHEN (c.population > '50000'::numeric) THEN COALESCE(counts.total_students, '0'::bigint) ELSE '0'::bigint END) DESC"

" Sort Method: quicksort Memory: 61kB"

" -> Hash Left Join (cost=82307.41..82321.15 rows=445 width=30) (actual time=11891.496..11891.758 rows=445 loops=1)"

"   Hash Cond: (c.id = counts.city_id)"

"   -> Seq Scan on ""Cities"" c (cost=0.00..11.45 rows=445 width=31) (actual time=0.029..0.117 rows=445 loops=1)"

"   -> Hash (cost=82307.06..82307.06 rows=28 width=12) (actual time=11891.458..11891.478 rows=28 loops=1)"

"   Buckets: 1024 Batches: 1 Memory Usage: 10kB"

"   -> Subquery Scan on counts (cost=82299.76..82307.06 rows=28 width=12) (actual time=11891.073..11891.468 rows=28 loops=1)"

"   -> GroupAggregate (cost=82299.76..82306.78 rows=28 width=12) (actual time=11891.072..11891.463 rows=28 loops=1)"

"   Group Key: p.city_id"

"   -> Sort (cost=82299.76..82302.01 rows=899 width=16) (actual time=11889.256..11889.306 rows=553 loops=1)"

"   Sort Key: p.city_id"

"   Sort Method: quicksort Memory: 50kB"

"   -> Nested Loop (cost=77798.33..82255.65 rows=899 width=16) (actual time=11791.769..11888.960 rows=553 loops=1)"

"   -> Nested Loop (cost=77797.90..81652.48 rows=899 width=24) (actual time=11791.710..11867.620 rows=553 loops=1)"

"   -> GroupAggregate (cost=77797.47..77851.43 rows=899 width=12) (actual time=11791.658..11858.889 rows=553 loops=1)"

"   Group Key: js.""StudentAMKA"""

"   Filter: (count(js.""ProgramID"") >= 2)"

"   Rows Removed by Filter: 58972"

"   -> Sort (cost=77797.47..77804.22 rows=2698 width=16) (actual time=11791.620..11828.664 rows=60081 loops=1)"

"   Sort Key: js.""StudentAMKA"""

"   Sort Method: external merge Disk: 1536kB"

"   -> Nested Loop (cost=0.85..77643.72 rows=2698 width=16) (actual time=1084.766..11245.759 rows=60081 loops=1)"

"   -> Nested Loop (cost=0.43..46016.22 rows=49142 width=16) (actual time=0.045..1887.210 rows=540089 loops=1)"

" -> Seq Scan on ""Program"" pgr (cost=0.00..2.50 rows=1 width=4) (actual time=0.023..0.067 rows=11 loops=1)"

" Filter: (""Duration"" = 5)"

" Rows Removed by Filter: 29"

" -> Index Only Scan using ""Joins_pkey"" on ""Joins"" js (cost=0.43..45522.30 rows=49142 width=16) (actual time=0.086..161.775 rows=49099 loops=11)"

" Index Cond: (""ProgramID"" = pgr.""ProgramID"")"

" Heap Fetches: 0"

" -> Index Scan using idx_student_amka on ""Student"" (cost=0.43..0.64 rows=1 width=12) (actual time=0.017..0.017 rows=0 loops=540089)"

" Index Cond: ((amka)::text = (js.""StudentAMKA"")::text)"

" Filter: ((entry_date >= '2040-09-01'::date) AND (entry_date <= '2050-09-30'::date))"

" Rows Removed by Filter: 1"

" -> Index Only Scan using idx_student_amka on ""Student"" s (cost=0.43..4.22 rows=1 width=12) (actual time=0.015..0.015 rows=1 loops=553)"

" Index Cond: (amka = (js.""StudentAMKA"")::text)"

" Heap Fetches: 0"

" -> Index Scan using idx_person_amka on ""Person"" p (cost=0.43..0.67 rows=1 width=16) (actual time=0.038..0.038 rows=1 loops=553)"

" Index Cond: ((amka)::text = (s.amka)::text)"

"Planning Time: 1.582 ms"

"Execution Time: 11892.900 ms"

ΔΟΚΙΜΗ 3Η

"Sort (cost=82340.73..82341.84 rows=445 width=30) (actual time=11162.188..11162.220 rows=445 loops=1)"

" Sort Key: (CASE WHEN (c.population > '50000'::numeric) THEN COALESCE(counts.total_students, '0'::bigint) ELSE '0'::bigint END) DESC"

" Sort Method: quicksort Memory: 61kB"

" -> Hash Left Join (cost=82307.41..82321.15 rows=445 width=30) (actual time=11161.817..11162.117 rows=445 loops=1)"

"   Hash Cond: (c.id = counts.city_id)"

"   -> Seq Scan on ""Cities"" c (cost=0.00..11.45 rows=445 width=31) (actual time=0.018..0.113 rows=445 loops=1)"

"   -> Hash (cost=82307.06..82307.06 rows=28 width=12) (actual time=11161.789..11161.794 rows=28 loops=1)"

"   Buckets: 1024 Batches: 1 Memory Usage: 10kB"

"   -> Subquery Scan on counts (cost=82299.76..82307.06 rows=28 width=12) (actual time=11161.342..11161.785 rows=28 loops=1)"

"   -> GroupAggregate (cost=82299.76..82306.78 rows=28 width=12) (actual time=11161.341..11161.779 rows=28 loops=1)"

"   Group Key: p.city_id"

"   -> Sort (cost=82299.76..82302.01 rows=899 width=16) (actual time=11159.535..11159.570 rows=553 loops=1)"

"   Sort Key: p.city_id"

"   Sort Method: quicksort Memory: 50kB"

"   -> Nested Loop (cost=77798.33..82255.65 rows=899 width=16) (actual time=11059.756..11159.195 rows=553 loops=1)"

"   -> Nested Loop (cost=77797.90..81652.48 rows=899 width=24) (actual time=11059.692..11137.895 rows=553 loops=1)"

"   -> GroupAggregate (cost=77797.47..77851.43 rows=899 width=12) (actual time=11059.641..11128.808 rows=553 loops=1)"

"   Group Key: js.""StudentAMKA"""

"   Filter: (count(js.""ProgramID"") >= 2)"

"   Rows Removed by Filter: 58972"

"   -> Sort (cost=77797.47..77804.22 rows=2698 width=16) (actual time=11059.600..11096.873 rows=60081 loops=1)"

"   Sort Key: js.""StudentAMKA"""

"   Sort Method: external merge Disk: 1536kB"

"   -> Nested Loop (cost=0.85..77643.72 rows=2698 width=16) (actual time=1273.496..10361.850 rows=60081 loops=1)"

"   -> Nested Loop (cost=0.43..46016.22 rows=49142 width=16) (actual time=0.033..1664.374 rows=540089 loops=1)"

"   -> Seq Scan on ""Program"" pgr (cost=0.00..2.50 rows=1 width=4) (actual time=0.015..0.054 rows=11 loops=1)"

"   Filter: (""Duration"" = 5)"

"   Rows Removed by Filter: 29"

"   -> Index Only Scan using ""Joins_pkey"" on ""Joins"" js (cost=0.43..45522.30 rows=49142 width=16) (actual time=0.044..142.502 rows=49099 loops=11)"

"   Index Cond: (""ProgramID"" = pgr.""ProgramID"")"

"   Heap Fetches: 0"

"   -> Index Scan using idx_student_amka on ""Student"" (cost=0.43..0.64 rows=1 width=12) (actual time=0.016..0.016 rows=0 loops=540089)"

"   Index Cond: ((amka)::text = (js.""StudentAMKA"")::text)"

"   Filter: ((entry_date >= '2040-09-01'::date) AND (entry_date <= '2050-09-30'::date))"

"   Rows Removed by Filter: 1"

"   -> Index Only Scan using idx_student_amka on ""Student"" s (cost=0.43..4.22 rows=1 width=12) (actual time=0.015..0.015 rows=1 loops=553)"

"   Index Cond: (amka = (js.""StudentAMKA"")::text)"

"   Heap Fetches: 0"

"   -> Index Scan using idx_person_amka on ""Person"" p (cost=0.43..0.67 rows=1 width=16) (actual time=0.037..0.037 rows=1 loops=553)"

"   Index Cond: ((amka)::text = (s.amka)::text)"

"Planning Time: 1.642 ms"

"Execution Time: 11163.859 ms"

(Cities: id, population | Person: amka | Student: amka, entry_date)

ΔΟΚΙΜΉ 1Η

"Sort (cost=59069.55..59070.66 rows=445 width=30) (actual time=3967.312..3967.344 rows=445 loops=1)"

" Sort Key: (CASE WHEN (c.population > '50000'::numeric) THEN COALESCE(counts.total_students, '0'::bigint) ELSE '0'::bigint END) DESC"

" Sort Method: quicksort Memory: 61kB"

" -> Hash Left Join (cost=59036.23..59049.98 rows=445 width=30) (actual time=3966.956..3967.233 rows=445 loops=1)"

"   Hash Cond: (c.id = counts.city_id)"

"   -> Seq Scan on ""Cities"" c (cost=0.00..11.45 rows=445 width=31) (actual time=0.023..0.133 rows=445 loops=1)"

"   -> Hash (cost=59035.88..59035.88 rows=28 width=12) (actual time=3966.922..3966.928 rows=28 loops=1)"

"   Buckets: 1024 Batches: 1 Memory Usage: 10kB"

"   -> Subquery Scan on counts (cost=59028.58..59035.88 rows=28 width=12) (actual time=3966.428..3966.915 rows=28 loops=1)"

"   -> GroupAggregate (cost=59028.58..59035.60 rows=28 width=12) (actual time=3966.427..3966.909 rows=28 loops=1)"

"   Group Key: p.city_id"

"   -> Sort (cost=59028.58..59030.83 rows=899 width=16) (actual time=3963.604..3963.653 rows=553 loops=1)"

"   Sort Key: p.city_id"

"   Sort Method: quicksort Memory: 50kB"

"   -> Nested Loop (cost=54527.15..58984.48 rows=899 width=16) (actual time=3134.836..3960.727 rows=553 loops=1)"

"   -> Nested Loop (cost=54526.72..58381.30 rows=899 width=24) (actual time=3134.225..3440.923 rows=553 loops=1)"

"   -> GroupAggregate (cost=54526.29..54580.25 rows=899 width=12) (actual time=3133.184..3231.099 rows=553 loops=1)"

"   Group Key: js.""StudentAMKA"""

"   Filter: (count(js.""ProgramID"") >= 2)"

"   Rows Removed by Filter: 58972"

"   -> Sort (cost=54526.29..54533.04 rows=2698 width=16) (actual time=3133.145..3189.073 rows=60081 loops=1)"

"   Sort Key: js.""StudentAMKA"""

"   Sort Method: external merge Disk: 1536kB"

"   -> Hash Join (cost=46630.93..54372.54 rows=2698 width=16) (actual time=1293.435..2627.731 rows=60081 loops=1)"

"   Hash Cond: ((""Student"".amka)::text = (js.""StudentAMKA"")::text)"

" -> Index Scan using idx_student_entry_date on ""Student"" (cost=0.43..7317.85 rows=105922 width=12) (actual time=0.507..879.650 rows=108262 loops=1)"

" Index Cond: ((entry_date >= '2040-09-01'::date) AND (entry_date <= '2050-09-30'::date))"

" -> Hash (cost=46016.22..46016.22 rows=49142 width=16) (actual time=1281.240..1281.242 rows=540089 loops=1)"

" Buckets: 131072 (originally 65536) Batches: 16 (originally 1) Memory Usage: 3073kB"

" -> Nested Loop (cost=0.43..46016.22 rows=49142 width=16) (actual time=0.066..1000.939 rows=540089 loops=1)"

" -> Seq Scan on ""Program"" pgr (cost=0.00..2.50 rows=1 width=4) (actual time=0.034..0.108 rows=11 loops=1)"

" Filter: (""Duration"" = 5)"

" Rows Removed by Filter: 29"

" -> Index Only Scan using ""Joins_pkey"" on ""Joins"" js (cost=0.43..45522.30 rows=49142 width=16) (actual time=0.014..83.461 rows=49099 loops=11)"

" Index Cond: (""ProgramID"" = pgr.""ProgramID"")"

" Heap Fetches: 0"

" -> Index Only Scan using idx_student_amka on ""Student"" s (cost=0.43..4.22 rows=1 width=12) (actual time=0.370..0.370 rows=1 loops=553)"

" Index Cond: (amka = (js.""StudentAMKA"")::text)"

" Heap Fetches: 0"

" -> Index Scan using idx_person_amka on ""Person"" p (cost=0.43..0.67 rows=1 width=16) (actual time=0.929..0.929 rows=1 loops=553)"

" Index Cond: ((amka)::text = (s.amka)::text)"

"**Planning Time: 11.397 ms"

"Execution Time: 3968.992 ms"

ΔΟΚΙΜΉ 2Η

"Sort (cost=59069.55..59070.66 rows=445 width=30) (actual time=2402.457..2402.487 rows=445 loops=1)"

" Sort Key: (CASE WHEN (c.population > '50000'::numeric) THEN COALESCE(counts.total_students, '0'::bigint) ELSE '0'::bigint END) DESC"

" Sort Method: quicksort Memory: 61kB"

" -> Hash Left Join (cost=59036.23..59049.98 rows=445 width=30) (actual time=2402.082..2402.318 rows=445 loops=1)"

"   Hash Cond: (c.id = counts.city_id)"

"   -> Seq Scan on ""Cities"" c (cost=0.00..11.45 rows=445 width=31) (actual time=0.031..0.106 rows=445 loops=1)"

"   -> Hash (cost=59035.88..59035.88 rows=28 width=12) (actual time=2402.037..2402.042 rows=28 loops=1)"

"   Buckets: 1024 Batches: 1 Memory Usage: 10kB"

"   -> Subquery Scan on counts (cost=59028.58..59035.88 rows=28 width=12) (actual time=2401.652..2402.031 rows=28 loops=1)"

"   -> GroupAggregate (cost=59028.58..59035.60 rows=28 width=12) (actual time=2401.651..2402.025 rows=28 loops=1)"

"   Group Key: p.city_id"

"   -> Sort (cost=59028.58..59030.83 rows=899 width=16) (actual time=2399.819..2399.855 rows=553 loops=1)"

"   Sort Key: p.city_id"

"   Sort Method: quicksort Memory: 50kB"

"   -> Nested Loop (cost=54527.15..58984.48 rows=899 width=16) (actual time=2316.151..2399.500 rows=553 loops=1)"

"   -> Nested Loop (cost=54526.72..58381.30 rows=899 width=24) (actual time=2316.124..2390.718 rows=553 loops=1)"

"   -> GroupAggregate (cost=54526.29..54580.25 rows=899 width=12) (actual time=2316.072..2382.646 rows=553 loops=1)"

"   Group Key: js.""StudentAMKA"""

"   Filter: (count(js.""ProgramID"") >= 2)"

"   Rows Removed by Filter: 58972"

"   -> Sort (cost=54526.29..54533.04 rows=2698 width=16) (actual time=2316.032..2353.436 rows=60081 loops=1)"

"   Sort Key: js.""StudentAMKA"""

"   Sort Method: external merge Disk: 1536kB"

"   -> Hash Join (cost=46630.93..54372.54 rows=2698 width=16) (actual time=1264.546..1773.251 rows=60081 loops=1)"

"   Hash Cond: (((""Student"".amka)::text = (js.""StudentAMKA"")::text)"

"   -> Index Scan using idx_student_entry_date on ""Student"" (cost=0.43..7317.85 rows=105922 width=12) (actual time=0.063..78.861 rows=108262 loops=1)"

" Index Cond: ((entry_date >= '2040-09-01'::date) AND (entry_date <= '2050-09-30'::date))"

" -> Hash (cost=46016.22..46016.22 rows=49142 width=16) (actual time=1259.972..1259.974 rows=540089 loops=1)"

" Buckets: 131072 (originally 65536) Batches: 16 (originally 1) Memory Usage: 3073kB"

" -> Nested Loop (cost=0.43..46016.22 rows=49142 width=16) (actual time=0.052..998.136 rows=540089 loops=1)"

" -> Seq Scan on ""Program"" pgr (cost=0.00..2.50 rows=1 width=4) (actual time=0.029..0.114 rows=11 loops=1)"

" Filter: (""Duration"" = 5)"

" Rows Removed by Filter: 29"

" -> Index Only Scan using ""Joins_pkey"" on ""Joins"" js (cost=0.43..45522.30 rows=49142 width=16) (actual time=0.013..82.545 rows=49099 loops=11)"

" Index Cond: (""ProgramID"" = pgr.""ProgramID"")"

" Heap Fetches: 0"

" -> Index Only Scan using idx_student_amka on ""Student"" s (cost=0.43..4.22 rows=1 width=12) (actual time=0.014..0.014 rows=1 loops=553)"

" Index Cond: (amka = (js.""StudentAMKA"")::text)"

" Heap Fetches: 0"

" -> Index Scan using idx_person_amka on ""Person"" p (cost=0.43..0.67 rows=1 width=16) (actual time=0.015..0.015 rows=1 loops=553)"

" Index Cond: ((amka)::text = (s.amka)::text)"

"Planning Time: 7.037 ms"

"Execution Time: 2403.909 ms"

ΔΟΚΙΜΉ 3Η

"Sort (cost=59069.55..59070.66 rows=445 width=30) (actual time=2373.127..2373.154 rows=445 loops=1)"

" Sort Key: (CASE WHEN (c.population > '50000'::numeric) THEN COALESCE(counts.total_students, '0'::bigint) ELSE '0'::bigint END) DESC"

" Sort Method: quicksort Memory: 61kB"

" -> Hash Left Join (cost=59036.23..59049.98 rows=445 width=30) (actual time=2372.843..2373.053 rows=445 loops=1)"

"   Hash Cond: (c.id = counts.city_id)"

"   -> Seq Scan on ""Cities"" c (cost=0.00..11.45 rows=445 width=31) (actual time=0.023..0.075 rows=445 loops=1)"

"   -> Hash (cost=59035.88..59035.88 rows=28 width=12) (actual time=2372.808..2372.812 rows=28 loops=1)"

"   Buckets: 1024 Batches: 1 Memory Usage: 10kB"

"   -> Subquery Scan on counts (cost=59028.58..59035.88 rows=28 width=12) (actual time=2372.428..2372.802 rows=28 loops=1)"

"   -> GroupAggregate (cost=59028.58..59035.60 rows=28 width=12) (actual time=2372.428..2372.797 rows=28 loops=1)"

"   Group Key: p.city_id"

"   -> Sort (cost=59028.58..59030.83 rows=899 width=16) (actual time=2370.684..2370.716 rows=553 loops=1)"

"   Sort Key: p.city_id"

"   Sort Method: quicksort Memory: 50kB"

"   -> Nested Loop (cost=54527.15..58984.48 rows=899 width=16) (actual time=2286.723..2370.477 rows=553 loops=1)"

"   -> Nested Loop (cost=54526.72..58381.30 rows=899 width=24) (actual time=2286.654..2362.420 rows=553 loops=1)"

"   -> GroupAggregate (cost=54526.29..54580.25 rows=899 width=12) (actual time=2286.553..2354.946 rows=553 loops=1)"

"   Group Key: js.""StudentAMKA"""

"   Filter: (count(js.""ProgramID"") >= 2)"

"   Rows Removed by Filter: 58972"

"   -> Sort (cost=54526.29..54533.04 rows=2698 width=16) (actual time=2286.515..2326.456 rows=60081 loops=1)"

"   Sort Key: js.""StudentAMKA"""

"   Sort Method: external merge Disk: 1536kB"

"   -> Hash Join (cost=46630.93..54372.54 rows=2698 width=16) (actual time=1200.688..1704.553 rows=60081 loops=1)"

"   Hash Cond: (((""Student"".amka)::text = (js.""StudentAMKA"")::text)"

"   -> Index Scan using idx_student_entry_date on ""Student"" (cost=0.43..7317.85 rows=105922 width=12) (actual time=0.028..80.435 rows=108262 loops=1)"

" Index Cond: ((entry_date >= '2040-09-01'::date) AND (entry_date <= '2050-09-30'::date))"

" -> Hash (cost=46016.22..46016.22 rows=49142 width=16) (actual time=1192.941..1192.943 rows=540089 loops=1)"

" Buckets: 131072 (originally 65536) Batches: 16 (originally 1) Memory Usage: 3073kB"

" -> Nested Loop (cost=0.43..46016.22 rows=49142 width=16) (actual time=0.029..913.365 rows=540089 loops=1)"

" -> Seq Scan on ""Program"" pgr (cost=0.00..2.50 rows=1 width=4) (actual time=0.011..0.063 rows=11 loops=1)"

" Filter: (""Duration"" = 5)"

" Rows Removed by Filter: 29"

" -> Index Only Scan using ""Joins_pkey"" on ""Joins"" js (cost=0.43..45522.30 rows=49142 width=16) (actual time=0.014..74.378 rows=49099 loops=11)"

" Index Cond: (""ProgramID"" = pgr.""ProgramID"")"

" Heap Fetches: 0"

" -> Index Only Scan using idx_student_amka on ""Student"" s (cost=0.43..4.22 rows=1 width=12) (actual time=0.013..0.013 rows=1 loops=553)"

" Index Cond: (amka = (js.""StudentAMKA"")::text)"

" Heap Fetches: 0"

" -> Index Scan using idx_person_amka on ""Person"" p (cost=0.43..0.67 rows=1 width=16) (actual time=0.014..0.014 rows=1 loops=553)"

" Index Cond: ((amka)::text = (s.amka)::text)"

"Planning Time: 0.891 ms"

"Execution Time: 2377.135 ms"

(Cities: id, population | Person: amka | Student: amka, entry_date | Joins: StudentAMKA)

ΔΟΚΙΜΉΗ 1Η

"Sort (cost=59069.55..59070.66 rows=445 width=30) (actual time=2403.500..2403.536 rows=445 loops=1)"

" Sort Key: (CASE WHEN (c.population > '50000'::numeric) THEN COALESCE(counts.total_students, '0'::bigint) ELSE '0'::bigint END) DESC"

" Sort Method: quicksort Memory: 61kB"

" -> Hash Left Join (cost=59036.23..59049.98 rows=445 width=30) (actual time=2402.993..2403.423 rows=445 loops=1)"

"   Hash Cond: (c.id = counts.city_id)"

"   -> Seq Scan on ""Cities"" c (cost=0.00..11.45 rows=445 width=31) (actual time=0.026..0.220 rows=445 loops=1)"

"   -> Hash (cost=59035.88..59035.88 rows=28 width=12) (actual time=2402.951..2402.961 rows=28 loops=1)"

"   Buckets: 1024 Batches: 1 Memory Usage: 10kB"

"   -> Subquery Scan on counts (cost=59028.58..59035.88 rows=28 width=12) (actual time=2402.024..2402.944 rows=28 loops=1)"

"   -> GroupAggregate (cost=59028.58..59035.60 rows=28 width=12) (actual time=2402.022..2402.936 rows=28 loops=1)"

"   Group Key: p.city_id"

"   -> Sort (cost=59028.58..59030.83 rows=899 width=16) (actual time=2399.523..2400.035 rows=553 loops=1)"

"   Sort Key: p.city_id"

"   Sort Method: quicksort Memory: 50kB"

"   -> Nested Loop (cost=54527.15..58984.48 rows=899 width=16) (actual time=2315.832..2399.312 rows=553 loops=1)"

"   -> Nested Loop (cost=54526.72..58381.30 rows=899 width=24) (actual time=2315.808..2390.734 rows=553 loops=1)"

"   -> GroupAggregate (cost=54526.29..54580.25 rows=899 width=12) (actual time=2315.760..2382.209 rows=553 loops=1)"

"   Group Key: js.""StudentAMKA"""

"   Filter: (count(js.""ProgramID"") >= 2)"

"   Rows Removed by Filter: 58972"

"   -> Sort (cost=54526.29..54533.04 rows=2698 width=16) (actual time=2315.704..2353.721 rows=60081 loops=1)"

"   Sort Key: js.""StudentAMKA"""

"   Sort Method: external merge Disk: 1536kB"

"   -> Hash Join (cost=46630.93..54372.54 rows=2698 width=16) (actual time=1296.667..1807.776 rows=60081 loops=1)"

"   Hash Cond: ((""Student"".amka)::text = (js.""StudentAMKA"")::text)"

" -> Index Scan using idx_student_entry_date on ""Student"" (cost=0.43..7317.85 rows=105922 width=12) (actual time=0.019..82.433 rows=108262 loops=1)"

" Index Cond: ((entry_date >= '2040-09-01'::date) AND (entry_date <= '2050-09-30'::date))"

" -> Hash (cost=46016.22..46016.22 rows=49142 width=16) (actual time=1292.307..1292.310 rows=540089 loops=1)"

" Buckets: 131072 (originally 65536) Batches: 16 (originally 1) Memory Usage: 3073kB"

" -> Nested Loop (cost=0.43..46016.22 rows=49142 width=16) (actual time=0.111..1032.181 rows=540089 loops=1)"

" -> Seq Scan on ""Program"" pgr (cost=0.00..2.50 rows=1 width=4) (actual time=0.011..0.061 rows=11 loops=1)"

" Filter: (""Duration"" = 5)"

" Rows Removed by Filter: 29"

" -> Index Only Scan using ""Joins_pkey"" on ""Joins"" js (cost=0.43..45522.30 rows=49142 width=16) (actual time=0.021..85.430 rows=49099 loops=11)"

" Index Cond: (""ProgramID"" = pgr.""ProgramID"")"

" Heap Fetches: 0"

" -> Index Only Scan using idx_student_amka on ""Student"" s (cost=0.43..4.22 rows=1 width=12) (actual time=0.015..0.015 rows=1 loops=553)"

" Index Cond: (amka = (js.""StudentAMKA"")::text)"

" Heap Fetches: 0"

" -> Index Scan using idx_person_amka on ""Person"" p (cost=0.43..0.67 rows=1 width=16) (actual time=0.015..0.015 rows=1 loops=553)"

" Index Cond: ((amka)::text = (s.amka)::text)"

"**Planning Time: 1.115 ms"

"Execution Time: 2405.421 ms"

ΔΟΚΙΜΉ 2Η

"Sort (cost=59069.55..59070.66 rows=445 width=30) (actual time=2353.284..2353.315 rows=445 loops=1)"

" Sort Key: (CASE WHEN (c.population > '50000'::numeric) THEN COALESCE(counts.total_students, '0'::bigint) ELSE '0'::bigint END) DESC"

" Sort Method: quicksort Memory: 61kB"

" -> Hash Left Join (cost=59036.23..59049.98 rows=445 width=30) (actual time=2352.994..2353.211 rows=445 loops=1)"

"   Hash Cond: (c.id = counts.city_id)"

"   -> Seq Scan on ""Cities"" c (cost=0.00..11.45 rows=445 width=31) (actual time=0.014..0.071 rows=445 loops=1)"

"   -> Hash (cost=59035.88..59035.88 rows=28 width=12) (actual time=2352.965..2352.972 rows=28 loops=1)"

"   Buckets: 1024 Batches: 1 Memory Usage: 10kB"

"   -> Subquery Scan on counts (cost=59028.58..59035.88 rows=28 width=12) (actual time=2352.426..2352.953 rows=28 loops=1)"

"   -> GroupAggregate (cost=59028.58..59035.60 rows=28 width=12) (actual time=2352.424..2352.944 rows=28 loops=1)"

"   Group Key: p.city_id"

"   -> Sort (cost=59028.58..59030.83 rows=899 width=16) (actual time=2345.755..2345.808 rows=553 loops=1)"

"   Sort Key: p.city_id"

"   Sort Method: quicksort Memory: 50kB"

"   -> Nested Loop (cost=54527.15..58984.48 rows=899 width=16) (actual time=2268.784..2345.583 rows=553 loops=1)"

"   -> Nested Loop (cost=54526.72..58381.30 rows=899 width=24) (actual time=2268.761..2338.478 rows=553 loops=1)"

"   -> GroupAggregate (cost=54526.29..54580.25 rows=899 width=12) (actual time=2268.712..2331.419 rows=553 loops=1)"

"   Group Key: js.""StudentAMKA"""

"   Filter: (count(js.""ProgramID"") >= 2)"

"   Rows Removed by Filter: 58972"

"   -> Sort (cost=54526.29..54533.04 rows=2698 width=16) (actual time=2268.673..2304.353 rows=60081 loops=1)"

"   Sort Key: js.""StudentAMKA"""

"   Sort Method: external merge Disk: 1536kB"

"   -> Hash Join (cost=46630.93..54372.54 rows=2698 width=16) (actual time=1253.479..1717.329 rows=60081 loops=1)"

"   Hash Cond: (((""Student"".amka)::text = (js.""StudentAMKA"")::text)"

"   -> Index Scan using idx_student_entry_date on ""Student"" (cost=0.43..7317.85 rows=105922 width=12) (actual time=0.015..74.602 rows=108262 loops=1)"

"   Index Cond: ((entry_date >= '2040-09-01'::date) AND (entry_date <= '2050-09-30'::date))"

"   -> Hash (cost=46016.22..46016.22 rows=49142 width=16) (actual time=1251.286..1251.288 rows=540089 loops=1)"

"   Buckets: 131072 (originally 65536) Batches: 16 (originally 1) Memory Usage: 3073kB"

"   -> Nested Loop (cost=0.43..46016.22 rows=49142 width=16) (actual time=0.024..974.032 rows=540089 loops=1)"

"   -> Seq Scan on ""Program"" pgr (cost=0.00..2.50 rows=1 width=4) (actual time=0.010..0.056 rows=11 loops=1)"

"   Filter: (""Duration"" = 5)"

"   Rows Removed by Filter: 29"

"   -> Index Only Scan using ""Joins_pkey"" on ""Joins"" js (cost=0.43..45522.30 rows=49142 width=16) (actual time=0.012..80.264 rows=49099 loops=11)"

"   Index Cond: (""ProgramID"" = pgr.""ProgramID"")"

"   Heap Fetches: 0"

"   -> Index Only Scan using idx_student_amka on ""Student"" s (cost=0.43..4.22 rows=1 width=12) (actual time=0.012..0.012 rows=1 loops=553)"

"   Index Cond: (amka = (js.""StudentAMKA"")::text)"

"   Heap Fetches: 0"

"   -> Index Scan using idx_person_amka on ""Person"" p (cost=0.43..0.67 rows=1 width=16) (actual time=0.012..0.012 rows=1 loops=553)"

"   Index Cond: ((amka)::text = (s.amka)::text)"

"Planning Time: 0.768 ms"

"Execution Time: 2354.685 ms"

ΔΟΚΙΜΉ 3Η

"Sort (cost=59069.55..59070.66 rows=445 width=30) (actual time=2847.272..2847.302 rows=445 loops=1)"

" Sort Key: (CASE WHEN (c.population > '50000'::numeric) THEN COALESCE(counts.total_students, '0'::bigint) ELSE '0'::bigint END) DESC"

" Sort Method: quicksort Memory: 61kB"

" -> Hash Left Join (cost=59036.23..59049.98 rows=445 width=30) (actual time=2846.996..2847.203 rows=445 loops=1)"

"   Hash Cond: (c.id = counts.city_id)"

"   -> Seq Scan on ""Cities"" c (cost=0.00..11.45 rows=445 width=31) (actual time=0.031..0.079 rows=445 loops=1)"

"   -> Hash (cost=59035.88..59035.88 rows=28 width=12) (actual time=2846.952..2846.956 rows=28 loops=1)"

"   Buckets: 1024 Batches: 1 Memory Usage: 10kB"

"   -> Subquery Scan on counts (cost=59028.58..59035.88 rows=28 width=12) (actual time=2846.574..2846.946 rows=28 loops=1)"

"   -> GroupAggregate (cost=59028.58..59035.60 rows=28 width=12) (actual time=2846.573..2846.941 rows=28 loops=1)"

"   Group Key: p.city_id"

"   -> Sort (cost=59028.58..59030.83 rows=899 width=16) (actual time=2844.827..2844.861 rows=553 loops=1)"

"   Sort Key: p.city_id"

"   Sort Method: quicksort Memory: 50kB"

"   -> Nested Loop (cost=54527.15..58984.48 rows=899 width=16) (actual time=2763.562..2844.607 rows=553 loops=1)"

"   -> Nested Loop (cost=54526.72..58381.30 rows=899 width=24) (actual time=2763.517..2835.204 rows=553 loops=1)"

"   -> GroupAggregate (cost=54526.29..54580.25 rows=899 width=12) (actual time=2763.260..2826.822 rows=553 loops=1)"

"   Group Key: js.""StudentAMKA"""

"   Filter: (count(js.""ProgramID"") >= 2)"

"   Rows Removed by Filter: 58972"

"   -> Sort (cost=54526.29..54533.04 rows=2698 width=16) (actual time=2763.220..2798.156 rows=60081 loops=1)"

"   Sort Key: js.""StudentAMKA"""

"   Sort Method: external merge Disk: 1536kB"

"   -> Hash Join (cost=46630.93..54372.54 rows=2698 width=16) (actual time=1591.770..2218.948 rows=60081 loops=1)"

"   Hash Cond: (((""Student"".amka)::text = (js.""StudentAMKA"")::text)"

"   -> Index Scan using idx_student_entry_date on ""Student"" (cost=0.43..7317.85 rows=105922 width=12) (actual time=0.015..97.284 rows=108262 loops=1)"

" Index Cond: ((entry_date >= '2040-09-01'::date) AND (entry_date <= '2050-09-30'::date))"

" -> Hash (cost=46016.22..46016.22 rows=49142 width=16) (actual time=1588.620..1588.622 rows=540089 loops=1)"

" Buckets: 131072 (originally 65536) Batches: 16 (originally 1) Memory Usage: 3073kB"

" -> Nested Loop (cost=0.43..46016.22 rows=49142 width=16) (actual time=0.025..1171.604 rows=540089 loops=1)"

" -> Seq Scan on ""Program"" pgr (cost=0.00..2.50 rows=1 width=4) (actual time=0.010..0.058 rows=11 loops=1)"

" Filter: (""Duration"" = 5)"

" Rows Removed by Filter: 29"

" -> Index Only Scan using ""Joins_pkey"" on ""Joins"" js (cost=0.43..45522.30 rows=49142 width=16) (actual time=0.014..96.184 rows=49099 loops=11)"

" Index Cond: (""ProgramID"" = pgr.""ProgramID"")"

" Heap Fetches: 0"

" -> Index Only Scan using idx_student_amka on ""Student"" s (cost=0.43..4.22 rows=1 width=12) (actual time=0.014..0.014 rows=1 loops=553)"

" Index Cond: (amka = (js.""StudentAMKA"")::text)"

" Heap Fetches: 0"

" -> Index Scan using idx_person_amka on ""Person"" p (cost=0.43..0.67 rows=1 width=16) (actual time=0.016..0.016 rows=1 loops=553)"

" Index Cond: ((amka)::text = (s.amka)::text)"

"Planning Time: 1.720 ms"

"Execution Time: 2850.509 ms"

(Cities: id, population | Person: amka | Student: amka, entry_date | Joins: StudentAMKA

Program: ProgramID)

ΔΟΚΙΜΉ 1Η

"Sort (cost=59069.55..59070.66 rows=445 width=30) (actual time=2011.527..2011.556 rows=445 loops=1)"

" Sort Key: (CASE WHEN (c.population > '50000'::numeric) THEN COALESCE(counts.total_students, '0'::bigint) ELSE '0'::bigint END) DESC"

" Sort Method: quicksort Memory: 61kB"

" -> Hash Left Join (cost=59036.23..59049.98 rows=445 width=30) (actual time=2011.246..2011.455 rows=445 loops=1)"

"   Hash Cond: (c.id = counts.city_id)"

"   -> Seq Scan on ""Cities"" c (cost=0.00..11.45 rows=445 width=31) (actual time=0.017..0.069 rows=445 loops=1)"

"   -> Hash (cost=59035.88..59035.88 rows=28 width=12) (actual time=2011.217..2011.222 rows=28 loops=1)"

"   Buckets: 1024 Batches: 1 Memory Usage: 10kB"

"   -> Subquery Scan on counts (cost=59028.58..59035.88 rows=28 width=12) (actual time=2010.836..2011.211 rows=28 loops=1)"

"   -> GroupAggregate (cost=59028.58..59035.60 rows=28 width=12) (actual time=2010.836..2011.206 rows=28 loops=1)"

"   Group Key: p.city_id"

"   -> Sort (cost=59028.58..59030.83 rows=899 width=16) (actual time=2009.087..2009.119 rows=553 loops=1)"

"   Sort Key: p.city_id"

"   Sort Method: quicksort Memory: 50kB"

"   -> Nested Loop (cost=54527.15..58984.48 rows=899 width=16) (actual time=1930.590..2008.890 rows=553 loops=1)"

"   -> Nested Loop (cost=54526.72..58381.30 rows=899 width=24) (actual time=1930.556..2001.154 rows=553 loops=1)"

"   -> GroupAggregate (cost=54526.29..54580.25 rows=899 width=12) (actual time=1930.502..1993.704 rows=553 loops=1)"

"   Group Key: js.""StudentAMKA"""

"   Filter: (count(js.""ProgramID"") >= 2)"

"   Rows Removed by Filter: 58972"

"   -> Sort (cost=54526.29..54533.04 rows=2698 width=16) (actual time=1930.458..1964.165 rows=60081 loops=1)"

"   Sort Key: js.""StudentAMKA"""

"   Sort Method: external merge Disk: 1536kB"

"   -> Hash Join (cost=46630.93..54372.54 rows=2698 width=16) (actual time=1013.059..1437.792 rows=60081 loops=1)"

" Hash Cond: (((""Student"".amka)::text = (js.""StudentAMKA"")::text)"

" -> Index Scan using idx_student_entry_date on ""Student"" (cost=0.43..7317.85 rows=105922 width=12) (actual time=0.019..69.691 rows=108262 loops=1)"

" Index Cond: ((entry_date >= '2040-09-01'::date) AND (entry_date <= '2050-09-30'::date))"

" -> Hash (cost=46016.22..46016.22 rows=49142 width=16) (actual time=1010.533..1010.534 rows=540089 loops=1)"

" Buckets: 131072 (originally 65536) Batches: 16 (originally 1) Memory Usage: 3073kB"

" -> Nested Loop (cost=0.43..46016.22 rows=49142 width=16) (actual time=0.028..794.291 rows=540089 loops=1)"

" -> Seq Scan on ""Program"" pgr (cost=0.00..2.50 rows=1 width=4) (actual time=0.010..0.055 rows=11 loops=1)"

" Filter: (""Duration"" = 5)"

" Rows Removed by Filter: 29"

" -> Index Only Scan using ""Joins_pkey"" on ""Joins"" js (cost=0.43..45522.30 rows=49142 width=16) (actual time=0.012..65.242 rows=49099 loops=11)"

" Index Cond: (""ProgramID"" = pgr.""ProgramID"")"

" Heap Fetches: 0"

" -> Index Only Scan using idx_student_amka on ""Student"" s (cost=0.43..4.22 rows=1 width=12) (actual time=0.013..0.013 rows=1 loops=553)"

" Index Cond: (amka = (js.""StudentAMKA"")::text)"

" Heap Fetches: 0"

" -> Index Scan using idx_person_amka on ""Person"" p (cost=0.43..0.67 rows=1 width=16) (actual time=0.013..0.013 rows=1 loops=553)"

" Index Cond: ((amka)::text = (s.amka)::text)"

"Planning Time: 2.221 ms"

"Execution Time: 2013.143 ms"

ΔΟΚΙΜΉ 2Η

"Sort (cost=59069.55..59070.66 rows=445 width=30) (actual time=2564.739..2564.768 rows=445 loops=1)"

" Sort Key: (CASE WHEN (c.population > '50000'::numeric) THEN COALESCE(counts.total_students, '0'::bigint) ELSE '0'::bigint END) DESC"

" Sort Method: quicksort Memory: 61kB"

" -> Hash Left Join (cost=59036.23..59049.98 rows=445 width=30) (actual time=2564.457..2564.666 rows=445 loops=1)"

"   Hash Cond: (c.id = counts.city_id)"

"   -> Seq Scan on ""Cities"" c (cost=0.00..11.45 rows=445 width=31) (actual time=0.019..0.069 rows=445 loops=1)"

"   -> Hash (cost=59035.88..59035.88 rows=28 width=12) (actual time=2564.427..2564.432 rows=28 loops=1)"

"   Buckets: 1024 Batches: 1 Memory Usage: 10kB"

"   -> Subquery Scan on counts (cost=59028.58..59035.88 rows=28 width=12) (actual time=2564.045..2564.421 rows=28 loops=1)"

"   -> GroupAggregate (cost=59028.58..59035.60 rows=28 width=12) (actual time=2564.045..2564.417 rows=28 loops=1)"

"   Group Key: p.city_id"

"   -> Sort (cost=59028.58..59030.83 rows=899 width=16) (actual time=2562.291..2562.326 rows=553 loops=1)"

"   Sort Key: p.city_id"

"   Sort Method: quicksort Memory: 50kB"

"   -> Nested Loop (cost=54527.15..58984.48 rows=899 width=16) (actual time=2453.201..2561.921 rows=553 loops=1)"

"   -> Nested Loop (cost=54526.72..58381.30 rows=899 width=24) (actual time=2453.160..2550.940 rows=553 loops=1)"

"   -> GroupAggregate (cost=54526.29..54580.25 rows=899 width=12) (actual time=2453.103..2537.923 rows=553 loops=1)"

"   Group Key: js.""StudentAMKA"""

"   Filter: (count(js.""ProgramID"") >= 2)"

"   Rows Removed by Filter: 58972"

"   -> Sort (cost=54526.29..54533.04 rows=2698 width=16) (actual time=2453.056..2498.670 rows=60081 loops=1)"

"   Sort Key: js.""StudentAMKA"""

"   Sort Method: external merge Disk: 1536kB"

"   -> Hash Join (cost=46630.93..54372.54 rows=2698 width=16) (actual time=1218.394..1848.739 rows=60081 loops=1)"

"   Hash Cond: (((""Student"".amka)::text = (js.""StudentAMKA"")::text)"

"   -> Index Scan using idx_student_entry_date on ""Student"" (cost=0.43..7317.85 rows=105922 width=12) (actual time=0.017..114.451 rows=108262 loops=1)"

" Index Cond: ((entry_date >= '2040-09-01'::date) AND (entry_date <= '2050-09-30'::date))"

" -> Hash (cost=46016.22..46016.22 rows=49142 width=16) (actual time=1216.513..1216.515 rows=540089 loops=1)"

" Buckets: 131072 (originally 65536) Batches: 16 (originally 1) Memory Usage: 3073kB"

" -> Nested Loop (cost=0.43..46016.22 rows=49142 width=16) (actual time=0.023..943.088 rows=540089 loops=1)"

" -> Seq Scan on ""Program"" pgr (cost=0.00..2.50 rows=1 width=4) (actual time=0.009..0.062 rows=11 loops=1)"

" Filter: (""Duration"" = 5)"

" Rows Removed by Filter: 29"

" -> Index Only Scan using ""Joins_pkey"" on ""Joins"" js (cost=0.43..45522.30 rows=49142 width=16) (actual time=0.017..77.890 rows=49099 loops=11)"

" Index Cond: (""ProgramID"" = pgr.""ProgramID"")"

" Heap Fetches: 0"

" -> Index Only Scan using idx_student_amka on ""Student"" s (cost=0.43..4.22 rows=1 width=12) (actual time=0.022..0.022 rows=1 loops=553)"

" Index Cond: (amka = (js.""StudentAMKA"")::text)"

" Heap Fetches: 0"

" -> Index Scan using idx_person_amka on ""Person"" p (cost=0.43..0.67 rows=1 width=16) (actual time=0.017..0.017 rows=1 loops=553)"

" Index Cond: ((amka)::text = (s.amka)::text)"

"Planning Time: 0.782 ms"

"Execution Time: 2567.679 ms"

ΔΟΚΙΜΉ 3Η

"Sort (cost=59069.55..59070.66 rows=445 width=30) (actual time=2809.272..2809.302 rows=445 loops=1)"

" Sort Key: (CASE WHEN (c.population > '50000'::numeric) THEN COALESCE(counts.total_students, '0'::bigint) ELSE '0'::bigint END) DESC"

" Sort Method: quicksort Memory: 61kB"

" -> Hash Left Join (cost=59036.23..59049.98 rows=445 width=30) (actual time=2808.993..2809.199 rows=445 loops=1)"

"   Hash Cond: (c.id = counts.city_id)"

"   -> Seq Scan on ""Cities"" c (cost=0.00..11.45 rows=445 width=31) (actual time=0.052..0.100 rows=445 loops=1)"

"   -> Hash (cost=59035.88..59035.88 rows=28 width=12) (actual time=2808.925..2808.929 rows=28 loops=1)"

"   Buckets: 1024 Batches: 1 Memory Usage: 10kB"

"   -> Subquery Scan on counts (cost=59028.58..59035.88 rows=28 width=12) (actual time=2808.542..2808.919 rows=28 loops=1)"

"   -> GroupAggregate (cost=59028.58..59035.60 rows=28 width=12) (actual time=2808.541..2808.914 rows=28 loops=1)"

"   Group Key: p.city_id"

"   -> Sort (cost=59028.58..59030.83 rows=899 width=16) (actual time=2806.817..2806.851 rows=553 loops=1)"

"   Sort Key: p.city_id"

"   Sort Method: quicksort Memory: 50kB"

"   -> Nested Loop (cost=54527.15..58984.48 rows=899 width=16) (actual time=2730.517..2806.609 rows=553 loops=1)"

"   -> Nested Loop (cost=54526.72..58381.30 rows=899 width=24) (actual time=2730.492..2798.890 rows=553 loops=1)"

"   -> GroupAggregate (cost=54526.29..54580.25 rows=899 width=12) (actual time=2730.446..2791.253 rows=553 loops=1)"

"   Group Key: js.""StudentAMKA"""

"   Filter: (count(js.""ProgramID"") >= 2)"

"   Rows Removed by Filter: 58972"

"   -> Sort (cost=54526.29..54533.04 rows=2698 width=16) (actual time=2730.406..2763.798 rows=60081 loops=1)"

"   Sort Key: js.""StudentAMKA"""

"   Sort Method: external merge Disk: 1536kB"

"   -> Hash Join (cost=46630.93..54372.54 rows=2698 width=16) (actual time=1190.092..1894.132 rows=60081 loops=1)"

"   Hash Cond: ((""Student"".amka)::text = (js.""StudentAMKA"")::text)"

"   -> Index Scan using idx_student_entry_date on ""Student"" (cost=0.43..7317.85 rows=105922 width=12) (actual time=0.019..209.223 rows=108262 loops=1)"

" Index Cond: ((entry_date >= '2040-09-01'::date) AND (entry_date <= '2050-09-30'::date))"

" -> Hash (cost=46016.22..46016.22 rows=49142 width=16) (actual time=1184.834..1184.836 rows=540089 loops=1)"

" Buckets: 131072 (originally 65536) Batches: 16 (originally 1) Memory Usage: 3073kB"

" -> Nested Loop (cost=0.43..46016.22 rows=49142 width=16) (actual time=0.039..935.773 rows=540089 loops=1)"

" -> Seq Scan on ""Program"" pgr (cost=0.00..2.50 rows=1 width=4) (actual time=0.016..0.067 rows=11 loops=1)"

" Filter: (""Duration"" = 5)"

" Rows Removed by Filter: 29"

" -> Index Only Scan using ""Joins_pkey"" on ""Joins"" js (cost=0.43..45522.30 rows=49142 width=16) (actual time=0.013..76.873 rows=49099 loops=11)"

" Index Cond: (""ProgramID"" = pgr.""ProgramID"")"

" Heap Fetches: 0"

" -> Index Only Scan using idx_student_amka on ""Student"" s (cost=0.43..4.22 rows=1 width=12) (actual time=0.013..0.013 rows=1 loops=553)"

" Index Cond: (amka = (js.""StudentAMKA"")::text)"

" Heap Fetches: 0"

" -> Index Scan using idx_person_amka on ""Person"" p (cost=0.43..0.67 rows=1 width=16) (actual time=0.013..0.013 rows=1 loops=553)"

" Index Cond: ((amka)::text = (s.amka)::text)"

"Planning Time: 0.780 ms"

"Execution Time: 2812.297 ms"

(Cities: id, population | Person: amka | Student: amka, entry_date | Joins: StudentAMKA

Program: ProgramID, Duration)

ΔΟΚΙΜΉΗ 1Η

"Sort (cost=59069.55..59070.66 rows=445 width=30) (actual time=1928.656..1928.719 rows=445 loops=1)"

" Sort Key: (CASE WHEN (c.population > '50000'::numeric) THEN COALESCE(counts.total_students, '0'::bigint) ELSE '0'::bigint END) DESC"

" Sort Method: quicksort Memory: 61kB"

" -> Hash Left Join (cost=59036.23..59049.98 rows=445 width=30) (actual time=1928.178..1928.537 rows=445 loops=1)"

"   Hash Cond: (c.id = counts.city_id)"

"   -> Seq Scan on ""Cities"" c (cost=0.00..11.45 rows=445 width=31) (actual time=0.019..0.103 rows=445 loops=1)"

"   -> Hash (cost=59035.88..59035.88 rows=28 width=12) (actual time=1928.146..1928.155 rows=28 loops=1)"

"   Buckets: 1024 Batches: 1 Memory Usage: 10kB"

"   -> Subquery Scan on counts (cost=59028.58..59035.88 rows=28 width=12) (actual time=1927.551..1928.073 rows=28 loops=1)"

"   -> GroupAggregate (cost=59028.58..59035.60 rows=28 width=12) (actual time=1927.550..1928.067 rows=28 loops=1)"

"   Group Key: p.city_id"

"   -> Sort (cost=59028.58..59030.83 rows=899 width=16) (actual time=1924.858..1924.960 rows=553 loops=1)"

"   Sort Key: p.city_id"

"   Sort Method: quicksort Memory: 50kB"

"   -> Nested Loop (cost=54527.15..58984.48 rows=899 width=16) (actual time=1845.208..1924.646 rows=553 loops=1)"

"   -> Nested Loop (cost=54526.72..58381.30 rows=899 width=24) (actual time=1845.182..1916.098 rows=553 loops=1)"

"   -> GroupAggregate (cost=54526.29..54580.25 rows=899 width=12) (actual time=1845.137..1908.297 rows=553 loops=1)"

"   Group Key: js.""StudentAMKA"""

"   Filter: (count(js.""ProgramID"") >= 2)"

"   Rows Removed by Filter: 58972"

"   -> Sort (cost=54526.29..54533.04 rows=2698 width=16) (actual time=1845.099..1878.362 rows=60081 loops=1)"

"   Sort Key: js.""StudentAMKA"""

"   Sort Method: external merge Disk: 1536kB"

"   -> Hash Join (cost=46630.93..54372.54 rows=2698 width=16) (actual time=976.260..1373.996 rows=60081 loops=1)"

" Hash Cond: (((""Student"".amka)::text = (js.""StudentAMKA"")::text)"

" -> Index Scan using idx_student_entry_date on ""Student"" (cost=0.43..7317.85 rows=105922 width=12) (actual time=0.015..60.563 rows=108262 loops=1)"

" Index Cond: ((entry_date >= '2040-09-01'::date) AND (entry_date <= '2050-09-30'::date))"

" -> Hash (cost=46016.22..46016.22 rows=49142 width=16) (actual time=973.443..973.446 rows=540089 loops=1)"

" Buckets: 131072 (originally 65536) Batches: 16 (originally 1) Memory Usage: 3073kB"

" -> Nested Loop (cost=0.43..46016.22 rows=49142 width=16) (actual time=0.025..766.665 rows=540089 loops=1)"

" -> Seq Scan on ""Program"" pgr (cost=0.00..2.50 rows=1 width=4) (actual time=0.010..0.058 rows=11 loops=1)"

" Filter: (""Duration"" = 5)"

" Rows Removed by Filter: 29"

" -> Index Only Scan using ""Joins_pkey"" on ""Joins"" js (cost=0.43..45522.30 rows=49142 width=16) (actual time=0.012..62.847 rows=49099 loops=11)"

" Index Cond: (""ProgramID"" = pgr.""ProgramID"")"

" Heap Fetches: 0"

" -> Index Only Scan using idx_student_amka on ""Student"" s (cost=0.43..4.22 rows=1 width=12) (actual time=0.013..0.013 rows=1 loops=553)"

" Index Cond: (amka = (js.""StudentAMKA"")::text)"

" Heap Fetches: 0"

" -> Index Scan using idx_person_amka on ""Person"" p (cost=0.43..0.67 rows=1 width=16) (actual time=0.014..0.014 rows=1 loops=553)"

" Index Cond: ((amka)::text = (s.amka)::text)"

"Planning Time: 2.525 ms"

"Execution Time: 1930.580 ms"

ΔΟΚΙΜΉ 2Η

"Sort (cost=59069.55..59070.66 rows=445 width=30) (actual time=2697.953..2697.982 rows=445 loops=1)"

" Sort Key: (CASE WHEN (c.population > '50000'::numeric) THEN COALESCE(counts.total_students, '0'::bigint) ELSE '0'::bigint END) DESC"

" Sort Method: quicksort Memory: 61kB"

" -> Hash Left Join (cost=59036.23..59049.98 rows=445 width=30) (actual time=2697.669..2697.880 rows=445 loops=1)"

"   Hash Cond: (c.id = counts.city_id)"

"   -> Seq Scan on ""Cities"" c (cost=0.00..11.45 rows=445 width=31) (actual time=0.029..0.079 rows=445 loops=1)"

"   -> Hash (cost=59035.88..59035.88 rows=28 width=12) (actual time=2697.626..2697.631 rows=28 loops=1)"

"   Buckets: 1024 Batches: 1 Memory Usage: 10kB"

"   -> Subquery Scan on counts (cost=59028.58..59035.88 rows=28 width=12) (actual time=2697.236..2697.622 rows=28 loops=1)"

"   -> GroupAggregate (cost=59028.58..59035.60 rows=28 width=12) (actual time=2697.236..2697.617 rows=28 loops=1)"

"   Group Key: p.city_id"

"   -> Sort (cost=59028.58..59030.83 rows=899 width=16) (actual time=2695.502..2695.536 rows=553 loops=1)"

"   Sort Key: p.city_id"

"   Sort Method: quicksort Memory: 50kB"

"   -> Nested Loop (cost=54527.15..58984.48 rows=899 width=16) (actual time=2618.751..2694.040 rows=553 loops=1)"

"   -> Nested Loop (cost=54526.72..58381.30 rows=899 width=24) (actual time=2618.726..2686.668 rows=553 loops=1)"

"   -> GroupAggregate (cost=54526.29..54580.25 rows=899 width=12) (actual time=2618.681..2678.347 rows=553 loops=1)"

"   Group Key: js.""StudentAMKA"""

"   Filter: (count(js.""ProgramID"") >= 2)"

"   Rows Removed by Filter: 58972"

"   -> Sort (cost=54526.29..54533.04 rows=2698 width=16) (actual time=2618.642..2651.729 rows=60081 loops=1)"

"   Sort Key: js.""StudentAMKA"""

"   Sort Method: external merge Disk: 1536kB"

"   -> Hash Join (cost=46630.93..54372.54 rows=2698 width=16) (actual time=1196.048..1999.527 rows=60081 loops=1)"

"   Hash Cond: (((""Student"".amka)::text = (js.""StudentAMKA"")::text)"

"   -> Index Scan using idx_student_entry_date on ""Student"" (cost=0.43..7317.85 rows=105922 width=12) (actual time=0.016..121.719 rows=108262 loops=1)"

"   Index Cond: ((entry_date >= '2040-09-01'::date) AND (entry_date <= '2050-09-30'::date))"

"   -> Hash (cost=46016.22..46016.22 rows=49142 width=16) (actual time=1192.758..1192.760 rows=540089 loops=1)"

"   Buckets: 131072 (originally 65536) Batches: 16 (originally 1) Memory Usage: 3073kB"

"   -> Nested Loop (cost=0.43..46016.22 rows=49142 width=16) (actual time=0.191..929.890 rows=540089 loops=1)"

"   -> Seq Scan on ""Program"" pgr (cost=0.00..2.50 rows=1 width=4) (actual time=0.096..0.143 rows=11 loops=1)"

"   Filter: (""Duration"" = 5)"

"   Rows Removed by Filter: 29"

"   -> Index Only Scan using ""Joins_pkey"" on ""Joins"" js (cost=0.43..45522.30 rows=49142 width=16) (actual time=0.019..76.796 rows=49099 loops=11)"

"   Index Cond: (""ProgramID"" = pgr.""ProgramID"")"

"   Heap Fetches: 0"

"   -> Index Only Scan using idx_student_amka on ""Student"" s (cost=0.43..4.22 rows=1 width=12) (actual time=0.014..0.014 rows=1 loops=553)"

"   Index Cond: (amka = (js.""StudentAMKA"")::text)"

"   Heap Fetches: 0"

"   -> Index Scan using idx_person_amka on ""Person"" p (cost=0.43..0.67 rows=1 width=16) (actual time=0.013..0.013 rows=1 loops=553)"

"   Index Cond: ((amka)::text = (s.amka)::text)"

"Planning Time: 1.382 ms"

"Execution Time: 2701.303 ms"

ΔΟΚΙΜΉ 3Η

"Sort (cost=59069.55..59070.66 rows=445 width=30) (actual time=2152.293..2152.322 rows=445 loops=1)"

" Sort Key: (CASE WHEN (c.population > '50000'::numeric) THEN COALESCE(counts.total_students, '0'::bigint) ELSE '0'::bigint END) DESC"

" Sort Method: quicksort Memory: 61kB"

" -> Hash Left Join (cost=59036.23..59049.98 rows=445 width=30) (actual time=2152.003..2152.219 rows=445 loops=1)"

"   Hash Cond: (c.id = counts.city_id)"

"   -> Seq Scan on ""Cities"" c (cost=0.00..11.45 rows=445 width=31) (actual time=0.012..0.068 rows=445 loops=1)"

"   -> Hash (cost=59035.88..59035.88 rows=28 width=12) (actual time=2151.980..2151.986 rows=28 loops=1)"

"   Buckets: 1024 Batches: 1 Memory Usage: 10kB"

"   -> Subquery Scan on counts (cost=59028.58..59035.88 rows=28 width=12) (actual time=2151.319..2151.972 rows=28 loops=1)"

"   -> GroupAggregate (cost=59028.58..59035.60 rows=28 width=12) (actual time=2151.318..2151.966 rows=28 loops=1)"

"   Group Key: p.city_id"

"   -> Sort (cost=59028.58..59030.83 rows=899 width=16) (actual time=2149.576..2149.618 rows=553 loops=1)"

"   Sort Key: p.city_id"

"   Sort Method: quicksort Memory: 50kB"

"   -> Nested Loop (cost=54527.15..58984.48 rows=899 width=16) (actual time=2075.932..2149.394 rows=553 loops=1)"

"   -> Nested Loop (cost=54526.72..58381.30 rows=899 width=24) (actual time=2075.909..2142.289 rows=553 loops=1)"

"   -> GroupAggregate (cost=54526.29..54580.25 rows=899 width=12) (actual time=2075.868..2135.216 rows=553 loops=1)"

"   Group Key: js.""StudentAMKA"""

"   Filter: (count(js.""ProgramID"") >= 2)"

"   Rows Removed by Filter: 58972"

"   -> Sort (cost=54526.29..54533.04 rows=2698 width=16) (actual time=2075.831..2108.774 rows=60081 loops=1)"

"   Sort Key: js.""StudentAMKA"""

"   Sort Method: external merge Disk: 1536kB"

"   -> Hash Join (cost=46630.93..54372.54 rows=2698 width=16) (actual time=1089.603..1559.004 rows=60081 loops=1)"

"   Hash Cond: (((""Student"".amka)::text = (js.""StudentAMKA"")::text)"

"   -> Index Scan using idx_student_entry_date on ""Student"" (cost=0.43..7317.85 rows=105922 width=12) (actual time=0.016..77.873 rows=108262 loops=1)"

"   Index Cond: ((entry_date >= '2040-09-01'::date) AND (entry_date <= '2050-09-30'::date))"

"   -> Hash (cost=46016.22..46016.22 rows=49142 width=16) (actual time=1087.322..1087.324 rows=540089 loops=1)"

"   Buckets: 131072 (originally 65536) Batches: 16 (originally 1) Memory Usage: 3073kB"

"   -> Nested Loop (cost=0.43..46016.22 rows=49142 width=16) (actual time=0.021..846.969 rows=540089 loops=1)"

"   -> Seq Scan on ""Program"" pgr (cost=0.00..2.50 rows=1 width=4) (actual time=0.008..0.117 rows=11 loops=1)"

"   Filter: (""Duration"" = 5)"

"   Rows Removed by Filter: 29"

"   -> Index Only Scan using ""Joins_pkey"" on ""Joins"" js (cost=0.43..45522.30 rows=49142 width=16) (actual time=0.011..69.712 rows=49099 loops=11)"

"   Index Cond: (""ProgramID"" = pgr.""ProgramID"")"

"   Heap Fetches: 0"

"   -> Index Only Scan using idx_student_amka on ""Student"" s (cost=0.43..4.22 rows=1 width=12) (actual time=0.012..0.012 rows=1 loops=553)"

"   Index Cond: (amka = (js.""StudentAMKA"")::text)"

"   Heap Fetches: 0"

"   -> Index Scan using idx_person_amka on ""Person"" p (cost=0.43..0.67 rows=1 width=16) (actual time=0.012..0.012 rows=1 loops=553)"

"   Index Cond: ((amka)::text = (s.amka)::text)"

"Planning Time: 1.209 ms"

"Execution Time: 2154.095 ms"

Query με χρήση ευρετηρίων HASH:

(Person: amka)

ΔΟΚΙΜΗ 1η

"Sort (cost=81957.83..81958.94 rows=445 width=30) (actual time=25168.360..25168.396 rows=445 loops=1)"

" Sort Key: (CASE WHEN (c.population > '50000'::numeric) THEN COALESCE(counts.total_students, '0'::bigint) ELSE '0'::bigint END) DESC"

" Sort Method: quicksort Memory: 61kB"

" -> Hash Left Join (cost=81924.51..81938.25 rows=445 width=30) (actual time=25166.379..25168.258 rows=445 loops=1)"

"   Hash Cond: (c.id = counts.city_id)"

"   -> Seq Scan on ""Cities"" c (cost=0.00..11.45 rows=445 width=31) (actual time=0.020..1.670 rows=445 loops=1)"

"   -> Hash (cost=81924.16..81924.16 rows=28 width=12) (actual time=25166.343..25166.350 rows=28 loops=1)"

"   Buckets: 1024 Batches: 1 Memory Usage: 10kB"

"   -> Subquery Scan on counts (cost=81916.86..81924.16 rows=28 width=12) (actual time=25165.942..25166.337 rows=28 loops=1)"

"   -> GroupAggregate (cost=81916.86..81923.88 rows=28 width=12) (actual time=25165.941..25166.331 rows=28 loops=1)"

"   Group Key: p.city_id"

"   -> Sort (cost=81916.86..81919.10 rows=899 width=16) (actual time=25164.173..25164.215 rows=553 loops=1)"

"   Sort Key: p.city_id"

"   Sort Method: quicksort Memory: 50kB"

"   -> Nested Loop (cost=77797.90..81872.75 rows=899 width=16) (actual time=24815.203..25162.719 rows=553 loops=1)"

"   -> Nested Loop (cost=77797.90..81652.48 rows=899 width=24) (actual time=24813.400..24900.255 rows=553 loops=1)"

"   -> GroupAggregate (cost=77797.47..77851.43 rows=899 width=12) (actual time=24812.839..24883.867 rows=553 loops=1)"

"   Group Key: js.""StudentAMKA"""

"   Filter: (count(js.""ProgramID"") >= 2)"

"   Rows Removed by Filter: 58972"

"   -> Sort (cost=77797.47..77804.22 rows=2698 width=16) (actual time=24812.800..24852.877 rows=60081 loops=1)"

"   Sort Key: js.""StudentAMKA"""

"   Sort Method: external merge Disk: 1536kB"

"   -> Nested Loop (cost=0.85..77643.72 rows=2698 width=16) (actual time=2458.761..24227.978 rows=60081 loops=1)"

" -> Nested Loop (cost=0.43..46016.22 rows=49142 width=16) (actual time=0.027..2006.020 rows=540089 loops=1)"

" -> Seq Scan on ""Program"" pgr (cost=0.00..2.50 rows=1 width=4) (actual time=0.010..0.432 rows=11 loops=1)"

" Filter: (""Duration"" = 5)"

" Rows Removed by Filter: 29"

" -> Index Only Scan using ""Joins_pkey"" on ""Joins"" js (cost=0.43..45522.30 rows=49142 width=16) (actual time=0.092..171.952 rows=49099 loops=11)"

" Index Cond: (""ProgramID"" = pgr.""ProgramID"")"

" Heap Fetches: 0"

" -> Index Scan using ""Student_pkey"" on ""Student"" (cost=0.43..0.64 rows=1 width=12) (actual time=0.040..0.040 rows=0 loops=540089)"

" Index Cond: ((amka)::text = (js.""StudentAMKA"")::text)"

" Filter: ((entry_date >= '2040-09-01'::date) AND (entry_date <= '2050-09-30'::date))"

" Rows Removed by Filter: 1"

" -> Index Only Scan using ""Student_pkey"" on ""Student"" s (cost=0.43..4.22 rows=1 width=12) (actual time=0.025..0.025 rows=1 loops=553)"

" Index Cond: (amka = (js.""StudentAMKA"")::text)"

" Heap Fetches: 0"

" -> Index Scan using idx_person_amka on ""Person"" p (cost=0.00..0.25 rows=1 width=16) (actual time=0.471..0.471 rows=1 loops=553)"

" Index Cond: ((amka)::text = (s.amka)::text)"

"Planning Time: 11.356 ms"

"Execution Time: 25169.362 ms"

ΔΟΚΙΜΉ 2Η

"Sort (cost=81957.83..81958.94 rows=445 width=30) (actual time=10657.141..10657.174 rows=445 loops=1)"

" Sort Key: (CASE WHEN (c.population > '50000'::numeric) THEN COALESCE(counts.total_students, '0'::bigint) ELSE '0'::bigint END) DESC"

" Sort Method: quicksort Memory: 61kB"

" -> Hash Left Join (cost=81924.51..81938.25 rows=445 width=30) (actual time=10656.601..10657.067 rows=445 loops=1)"

"   Hash Cond: (c.id = counts.city_id)"

"   -> Seq Scan on ""Cities"" c (cost=0.00..11.45 rows=445 width=31) (actual time=0.023..0.315 rows=445 loops=1)"

"   -> Hash (cost=81924.16..81924.16 rows=28 width=12) (actual time=10656.503..10656.510 rows=28 loops=1)"

"   Buckets: 1024 Batches: 1 Memory Usage: 10kB"

"   -> Subquery Scan on counts (cost=81916.86..81924.16 rows=28 width=12) (actual time=10656.102..10656.497 rows=28 loops=1)"

"   -> GroupAggregate (cost=81916.86..81923.88 rows=28 width=12) (actual time=10656.101..10656.490 rows=28 loops=1)"

"   Group Key: p.city_id"

"   -> Sort (cost=81916.86..81919.10 rows=899 width=16) (actual time=10654.015..10654.058 rows=553 loops=1)"

"   Sort Key: p.city_id"

"   Sort Method: quicksort Memory: 50kB"

"   -> Nested Loop (cost=77797.90..81872.75 rows=899 width=16) (actual time=10557.673..10653.691 rows=553 loops=1)"

"   -> Nested Loop (cost=77797.90..81652.48 rows=899 width=24) (actual time=10557.613..10634.226 rows=553 loops=1)"

"   -> GroupAggregate (cost=77797.47..77851.43 rows=899 width=12) (actual time=10557.550..10623.991 rows=553 loops=1)"

"   Group Key: js.""StudentAMKA"""

"   Filter: (count(js.""ProgramID"") >= 2)"

"   Rows Removed by Filter: 58972"

"   -> Sort (cost=77797.47..77804.22 rows=2698 width=16) (actual time=10557.507..10594.803 rows=60081 loops=1)"

"   Sort Key: js.""StudentAMKA"""

"   Sort Method: external merge Disk: 1536kB"

"   -> Nested Loop (cost=0.85..77643.72 rows=2698 width=16) (actual time=1062.235..9972.009 rows=60081 loops=1)"

"   -> Nested Loop (cost=0.43..46016.22 rows=49142 width=16) (actual time=2.717..1597.237 rows=540089 loops=1)"

"   -> Seq Scan on ""Program"" pgr (cost=0.00..2.50 rows=1 width=4) (actual time=2.645..2.682 rows=11 loops=1)"

"   Filter: (""Duration"" = 5)"

"   Rows Removed by Filter: 29"

"   -> Index Only Scan using ""Joins_pkey"" on ""Joins"" js (cost=0.43..45522.30 rows=49142 width=16) (actual time=0.034..136.511 rows=49099 loops=11)"

"   Index Cond: (""ProgramID"" = pgr.""ProgramID"")"

"   Heap Fetches: 0"

"   -> Index Scan using ""Student_pkey"" on ""Student"" (cost=0.43..0.64 rows=1 width=12) (actual time=0.015..0.015 rows=0 loops=540089)"

"   Index Cond: ((amka)::text = (js.""StudentAMKA"")::text)"

"   Filter: ((entry_date >= '2040-09-01'::date) AND (entry_date <= '2050-09-30'::date))"

"   Rows Removed by Filter: 1"

"   -> Index Only Scan using ""Student_pkey"" on ""Student"" s (cost=0.43..4.22 rows=1 width=12) (actual time=0.017..0.017 rows=1 loops=553)"

"   Index Cond: (amka = (js.""StudentAMKA"")::text)"

"   Heap Fetches: 0"

"   -> Index Scan using idx_person_amka on ""Person"" p (cost=0.00..0.25 rows=1 width=16) (actual time=0.033..0.033 rows=1 loops=553)"

"   Index Cond: ((amka)::text = (s.amka)::text)"

"Planning Time: 6.782 ms"

"Execution Time: 10658.248 ms"

ΔΟΚΙΜΉ 3Η

"Sort (cost=81957.83..81958.94 rows=445 width=30) (actual time=10945.100..10945.127 rows=445 loops=1)"

" Sort Key: (CASE WHEN (c.population > '50000'::numeric) THEN COALESCE(counts.total_students, '0'::bigint) ELSE '0'::bigint END) DESC"

" Sort Method: quicksort Memory: 61kB"

" -> Hash Left Join (cost=81924.51..81938.25 rows=445 width=30) (actual time=10944.795..10945.030 rows=445 loops=1)"

"   Hash Cond: (c.id = counts.city_id)"

"   -> Seq Scan on ""Cities"" c (cost=0.00..11.45 rows=445 width=31) (actual time=0.016..0.091 rows=445 loops=1)"

"   -> Hash (cost=81924.16..81924.16 rows=28 width=12) (actual time=10944.771..10944.774 rows=28 loops=1)"

"   Buckets: 1024 Batches: 1 Memory Usage: 10kB"

"   -> Subquery Scan on counts (cost=81916.86..81924.16 rows=28 width=12) (actual time=10944.385..10944.764 rows=28 loops=1)"

"   -> GroupAggregate (cost=81916.86..81923.88 rows=28 width=12) (actual time=10944.384..10944.759 rows=28 loops=1)"

"   Group Key: p.city_id"

"   -> Sort (cost=81916.86..81919.10 rows=899 width=16) (actual time=10942.624..10942.657 rows=553 loops=1)"

"   Sort Key: p.city_id"

"   Sort Method: quicksort Memory: 50kB"

"   -> Nested Loop (cost=77797.90..81872.75 rows=899 width=16) (actual time=10850.109..10942.363 rows=553 loops=1)"

"   -> Nested Loop (cost=77797.90..81652.48 rows=899 width=24) (actual time=10850.060..10927.411 rows=553 loops=1)"

"   -> GroupAggregate (cost=77797.47..77851.43 rows=899 width=12) (actual time=10850.006..10919.124 rows=553 loops=1)"

"   Group Key: js.""StudentAMKA"""

"   Filter: (count(js.""ProgramID"") >= 2)"

"   Rows Removed by Filter: 58972"

"   -> Sort (cost=77797.47..77804.22 rows=2698 width=16) (actual time=10849.969..10888.580 rows=60081 loops=1)"

"   Sort Key: js.""StudentAMKA"""

"   Sort Method: external merge Disk: 1536kB"

"   -> Nested Loop (cost=0.85..77643.72 rows=2698 width=16) (actual time=952.353..10232.280 rows=60081 loops=1)"

"   -> Nested Loop (cost=0.43..46016.22 rows=49142 width=16) (actual time=0.034..1703.563 rows=540089 loops=1)"

" -> Seq Scan on ""Program"" pgr (cost=0.00..2.50 rows=1 width=4) (actual time=0.015..0.056 rows=11 loops=1)"

" Filter: (""Duration"" = 5)"

" Rows Removed by Filter: 29"

" -> Index Only Scan using ""Joins_pkey"" on ""Joins"" js (cost=0.43..45522.30 rows=49142 width=16) (actual time=0.035..145.623 rows=49099 loops=11)"

" Index Cond: (""ProgramID"" = pgr.""ProgramID"")"

" Heap Fetches: 0"

" -> Index Scan using ""Student_pkey"" on ""Student"" (cost=0.43..0.64 rows=1 width=12) (actual time=0.015..0.015 rows=0 loops=540089)"

" Index Cond: ((amka)::text = (js.""StudentAMKA"")::text)"

" Filter: ((entry_date >= '2040-09-01'::date) AND (entry_date <= '2050-09-30'::date))"

" Rows Removed by Filter: 1"

" -> Index Only Scan using ""Student_pkey"" on ""Student"" s (cost=0.43..4.22 rows=1 width=12) (actual time=0.014..0.014 rows=1 loops=553)"

" Index Cond: (amka = (js.""StudentAMKA"")::text)"

" Heap Fetches: 0"

" -> Index Scan using idx_person_amka on ""Person"" p (cost=0.00..0.25 rows=1 width=16) (actual time=0.026..0.026 rows=1 loops=553)"

" Index Cond: ((amka)::text = (s.amka)::text)"

"Planning Time: 1.503 ms"

"Execution Time: 10946.027 ms"

(Person: amka | Student: amka)

ΔΟΚΙΜΗ 1η

"Sort (cost=61025.82..61026.93 rows=445 width=30) (actual time=15292.413..15292.443 rows=445 loops=1)"

" Sort Key: (CASE WHEN (c.population > '50000'::numeric) THEN COALESCE(counts.total_students, '0'::bigint) ELSE '0'::bigint END) DESC"

" Sort Method: quicksort Memory: 61kB"

" -> Hash Left Join (cost=60992.50..61006.25 rows=445 width=30) (actual time=15292.102..15292.337 rows=445 loops=1)"

"   Hash Cond: (c.id = counts.city_id)"

"   -> Seq Scan on ""Cities"" c (cost=0.00..11.45 rows=445 width=31) (actual time=0.021..0.095 rows=445 loops=1)"

"   -> Hash (cost=60992.15..60992.15 rows=28 width=12) (actual time=15292.005..15292.008 rows=28 loops=1)"

"   Buckets: 1024 Batches: 1 Memory Usage: 10kB"

"   -> Subquery Scan on counts (cost=60984.85..60992.15 rows=28 width=12) (actual time=15291.622..15291.999 rows=28 loops=1)"

"   -> GroupAggregate (cost=60984.85..60991.87 rows=28 width=12) (actual time=15291.622..15291.993 rows=28 loops=1)"

"   Group Key: p.city_id"

"   -> Sort (cost=60984.85..60987.10 rows=899 width=16) (actual time=15289.367..15289.400 rows=553 loops=1)"

"   Sort Key: p.city_id"

"   Sort Method: quicksort Memory: 50kB"

"   -> Nested Loop (cost=56865.89..60940.75 rows=899 width=16) (actual time=14647.309..15287.519 rows=553 loops=1)"

"   -> Nested Loop (cost=56865.89..60720.47 rows=899 width=24) (actual time=14646.392..14935.726 rows=553 loops=1)"

"   -> GroupAggregate (cost=56865.47..56919.43 rows=899 width=12) (actual time=14644.400..14715.003 rows=553 loops=1)"

"   Group Key: js.""StudentAMKA"""

"   Filter: (count(js.""ProgramID"") >= 2)"

"   Rows Removed by Filter: 58972"

"   -> Sort (cost=56865.47..56872.21 rows=2698 width=16) (actual time=14644.357..14684.191 rows=60081 loops=1)"

"   Sort Key: js.""StudentAMKA"""

"   Sort Method: external merge Disk: 1536kB"

"   -> Nested Loop (cost=0.43..56711.71 rows=2698 width=16) (actual time=1311.994..14088.397 rows=60081 loops=1)"

"   -> Nested Loop (cost=0.43..46016.22 rows=49142 width=16) (actual time=0.062..1695.060 rows=540089 loops=1)"

"   -> Seq Scan on ""Program"" pgr (cost=0.00..2.50 rows=1 width=4) (actual time=0.025..0.071 rows=11 loops=1)"

"   Filter: (""Duration"" = 5)"

"   Rows Removed by Filter: 29"

"   -> Index Only Scan using ""Joins_pkey"" on ""Joins"" js (cost=0.43..45522.30 rows=49142 width=16) (actual time=0.033..145.341 rows=49099 loops=11)"

"   Index Cond: (""ProgramID"" = pgr.""ProgramID"")"

"   Heap Fetches: 0"

"   -> Index Scan using idx_student_amka on ""Student"" (cost=0.00..0.22 rows=1 width=12) (actual time=0.022..0.022 rows=0 loops=540089)"

"   Index Cond: ((amka)::text = (js.""StudentAMKA"")::text)"

"   Rows Removed by Index Recheck: 0"

"   Filter: ((entry_date >= '2040-09-01'::date) AND (entry_date <= '2050-09-30'::date))"

"   Rows Removed by Filter: 1"

"   -> Index Only Scan using ""Student_pkey"" on ""Student"" s (cost=0.43..4.22 rows=1 width=12) (actual time=0.393..0.393 rows=1 loops=553)"

"   Index Cond: (amka = (js.""StudentAMKA"")::text)"

"   Heap Fetches: 0"

"   -> Index Scan using idx_person_amka on ""Person"" p (cost=0.00..0.25 rows=1 width=16) (actual time=0.630..0.630 rows=1 loops=553)"

"   Index Cond: ((amka)::text = (s.amka)::text)"

"Planning Time: 14.938 ms"

"Execution Time: 15293.413 ms"

ΔΟΚΙΜΉ 2Η

"Sort (cost=61025.82..61026.93 rows=445 width=30) (actual time=15292.413..15292.443 rows=445 loops=1)"

" Sort Key: (CASE WHEN (c.population > '50000'::numeric) THEN COALESCE(counts.total_students, '0'::bigint) ELSE '0'::bigint END) DESC"

" Sort Method: quicksort Memory: 61kB"

" -> Hash Left Join (cost=60992.50..61006.25 rows=445 width=30) (actual time=15292.102..15292.337 rows=445 loops=1)"

"   Hash Cond: (c.id = counts.city_id)"

"   -> Seq Scan on ""Cities"" c (cost=0.00..11.45 rows=445 width=31) (actual time=0.021..0.095 rows=445 loops=1)"

"   -> Hash (cost=60992.15..60992.15 rows=28 width=12) (actual time=15292.005..15292.008 rows=28 loops=1)"

"   Buckets: 1024 Batches: 1 Memory Usage: 10kB"

"   -> Subquery Scan on counts (cost=60984.85..60992.15 rows=28 width=12) (actual time=15291.622..15291.999 rows=28 loops=1)"

"   -> GroupAggregate (cost=60984.85..60991.87 rows=28 width=12) (actual time=15291.622..15291.993 rows=28 loops=1)"

"   Group Key: p.city_id"

"   -> Sort (cost=60984.85..60987.10 rows=899 width=16) (actual time=15289.367..15289.400 rows=553 loops=1)"

"   Sort Key: p.city_id"

"   Sort Method: quicksort Memory: 50kB"

"   -> Nested Loop (cost=56865.89..60940.75 rows=899 width=16) (actual time=14647.309..15287.519 rows=553 loops=1)"

"   -> Nested Loop (cost=56865.89..60720.47 rows=899 width=24) (actual time=14646.392..14935.726 rows=553 loops=1)"

"   -> GroupAggregate (cost=56865.47..56919.43 rows=899 width=12) (actual time=14644.400..14715.003 rows=553 loops=1)"

"   Group Key: js.""StudentAMKA"""

"   Filter: (count(js.""ProgramID"") >= 2)"

"   Rows Removed by Filter: 58972"

"   -> Sort (cost=56865.47..56872.21 rows=2698 width=16) (actual time=14644.357..14684.191 rows=60081 loops=1)"

"   Sort Key: js.""StudentAMKA"""

"   Sort Method: external merge Disk: 1536kB"

"   -> Nested Loop (cost=0.43..56711.71 rows=2698 width=16) (actual time=1311.994..14088.397 rows=60081 loops=1)"

"   -> Nested Loop (cost=0.43..46016.22 rows=49142 width=16) (actual time=0.062..1695.060 rows=540089 loops=1)"

" -> Seq Scan on ""Program"" pgr (cost=0.00..2.50 rows=1 width=4) (actual time=0.025..0.071 rows=11 loops=1)"

" Filter: (""Duration"" = 5)"

" Rows Removed by Filter: 29"

" -> Index Only Scan using ""Joins_pkey"" on ""Joins"" js (cost=0.43..45522.30 rows=49142 width=16) (actual time=0.033..145.341 rows=49099 loops=11)"

" Index Cond: (""ProgramID"" = pgr.""ProgramID"")"

" Heap Fetches: 0"

" -> Index Scan using idx_student_amka on ""Student"" (cost=0.00..0.22 rows=1 width=12) (actual time=0.022..0.022 rows=0 loops=540089)"

" Index Cond: ((amka)::text = (js.""StudentAMKA"")::text)"

" Rows Removed by Index Recheck: 0"

" Filter: ((entry_date >= '2040-09-01'::date) AND (entry_date <= '2050-09-30'::date))"

" Rows Removed by Filter: 1"

" -> Index Only Scan using ""Student_pkey"" on ""Student"" s (cost=0.43..4.22 rows=1 width=12) (actual time=0.393..0.393 rows=1 loops=553)"

" Index Cond: (amka = (js.""StudentAMKA"")::text)"

" Heap Fetches: 0"

" -> Index Scan using idx_person_amka on ""Person"" p (cost=0.00..0.25 rows=1 width=16) (actual time=0.630..0.630 rows=1 loops=553)"

" Index Cond: ((amka)::text = (s.amka)::text)"

"Planning Time: 14.938 ms"

"Execution Time: 15293.413 ms"

ΔΟΚΙΜΉ 2Η

"Sort (cost=61025.82..61026.93 rows=445 width=30) (actual time=6991.243..6991.283 rows=445 loops=1)"

" Sort Key: (CASE WHEN (c.population > '50000'::numeric) THEN COALESCE(counts.total_students, '0'::bigint) ELSE '0'::bigint END) DESC"

" Sort Method: quicksort Memory: 61kB"

" -> Hash Left Join (cost=60992.50..61006.25 rows=445 width=30) (actual time=6990.630..6991.002 rows=445 loops=1)"

"   Hash Cond: (c.id = counts.city_id)"

"   -> Seq Scan on ""Cities"" c (cost=0.00..11.45 rows=445 width=31) (actual time=0.019..0.188 rows=445 loops=1)"

"   -> Hash (cost=60992.15..60992.15 rows=28 width=12) (actual time=6990.488..6990.501 rows=28 loops=1)"

"   Buckets: 1024 Batches: 1 Memory Usage: 10kB"

"   -> Subquery Scan on counts (cost=60984.85..60992.15 rows=28 width=12) (actual time=6989.910..6990.481 rows=28 loops=1)"

"   -> GroupAggregate (cost=60984.85..60991.87 rows=28 width=12) (actual time=6989.908..6990.474 rows=28 loops=1)"

"   Group Key: p.city_id"

"   -> Sort (cost=60984.85..60987.10 rows=899 width=16) (actual time=6987.775..6987.841 rows=553 loops=1)"

"   Sort Key: p.city_id"

"   Sort Method: quicksort Memory: 50kB"

"   -> Nested Loop (cost=56865.89..60940.75 rows=899 width=16) (actual time=6892.343..6987.469 rows=553 loops=1)"

"   -> Nested Loop (cost=56865.89..60720.47 rows=899 width=24) (actual time=6892.265..6971.611 rows=553 loops=1)"

"   -> GroupAggregate (cost=56865.47..56919.43 rows=899 width=12) (actual time=6892.135..6957.910 rows=553 loops=1)"

"   Group Key: js.""StudentAMKA"""

"   Filter: (count(js.""ProgramID"") >= 2)"

"   Rows Removed by Filter: 58972"

"   -> Sort (cost=56865.47..56872.21 rows=2698 width=16) (actual time=6892.085..6928.057 rows=60081 loops=1)"

"   Sort Key: js.""StudentAMKA"""

"   Sort Method: external merge Disk: 1536kB"

"   -> Nested Loop (cost=0.43..56711.71 rows=2698 width=16) (actual time=1396.538..6278.612 rows=60081 loops=1)"

"   -> Nested Loop (cost=0.43..46016.22 rows=49142 width=16) (actual time=0.036..1699.545 rows=540089 loops=1)"

" -> Seq Scan on ""Program"" pgr (cost=0.00..2.50 rows=1 width=4) (actual time=0.016..0.054 rows=11 loops=1)"

" Filter: (""Duration"" = 5)"

" Rows Removed by Filter: 29"

" -> Index Only Scan using ""Joins_pkey"" on ""Joins"" js (cost=0.43..45522.30 rows=49142 width=16) (actual time=0.043..145.384 rows=49099 loops=11)"

" Index Cond: (""ProgramID"" = pgr.""ProgramID"")"

" Heap Fetches: 0"

" -> Index Scan using idx_student_amka on ""Student"" (cost=0.00..0.22 rows=1 width=12) (actual time=0.008..0.008 rows=0 loops=540089)"

" Index Cond: ((amka)::text = (js.""StudentAMKA"")::text)"

" Rows Removed by Index Recheck: 0"

" Filter: ((entry_date >= '2040-09-01'::date) AND (entry_date <= '2050-09-30'::date))"

" Rows Removed by Filter: 1"

" -> Index Only Scan using ""Student_pkey"" on ""Student"" s (cost=0.43..4.22 rows=1 width=12) (actual time=0.024..0.024 rows=1 loops=553)"

" Index Cond: (amka = (js.""StudentAMKA"")::text)"

" Heap Fetches: 0"

" -> Index Scan using idx_person_amka on ""Person"" p (cost=0.00..0.25 rows=1 width=16) (actual time=0.028..0.028 rows=1 loops=553)"

" Index Cond: ((amka)::text = (s.amka)::text)"

"Planning Time: 2.131 ms"

"Execution Time: 6994.075 ms"

ΔΟΚΙΜΉΗ 3Η

"Sort (cost=61025.82..61026.93 rows=445 width=30) (actual time=5220.030..5220.060 rows=445 loops=1)"

" Sort Key: (CASE WHEN (c.population > '50000'::numeric) THEN COALESCE(counts.total_students, '0'::bigint) ELSE '0'::bigint END) DESC"

" Sort Method: quicksort Memory: 61kB"

" -> Hash Left Join (cost=60992.50..61006.25 rows=445 width=30) (actual time=5219.688..5219.956 rows=445 loops=1)"

"   Hash Cond: (c.id = counts.city_id)"

"   -> Seq Scan on ""Cities"" c (cost=0.00..11.45 rows=445 width=31) (actual time=0.014..0.121 rows=445 loops=1)"

"   -> Hash (cost=60992.15..60992.15 rows=28 width=12) (actual time=5219.661..5219.666 rows=28 loops=1)"

"   Buckets: 1024 Batches: 1 Memory Usage: 10kB"

"   -> Subquery Scan on counts (cost=60984.85..60992.15 rows=28 width=12) (actual time=5219.195..5219.646 rows=28 loops=1)"

"   -> GroupAggregate (cost=60984.85..60991.87 rows=28 width=12) (actual time=5219.194..5219.640 rows=28 loops=1)"

"   Group Key: p.city_id"

"   -> Sort (cost=60984.85..60987.10 rows=899 width=16) (actual time=5216.695..5216.736 rows=553 loops=1)"

"   Sort Key: p.city_id"

"   Sort Method: quicksort Memory: 50kB"

"   -> Nested Loop (cost=56865.89..60940.75 rows=899 width=16) (actual time=5116.440..5216.333 rows=553 loops=1)"

"   -> Nested Loop (cost=56865.89..60720.47 rows=899 width=24) (actual time=5116.374..5200.548 rows=553 loops=1)"

"   -> GroupAggregate (cost=56865.47..56919.43 rows=899 width=12) (actual time=5116.230..5185.301 rows=553 loops=1)"

"   Group Key: js.""StudentAMKA"""

"   Filter: (count(js.""ProgramID"") >= 2)"

"   Rows Removed by Filter: 58972"

"   -> Sort (cost=56865.47..56872.21 rows=2698 width=16) (actual time=5116.192..5154.116 rows=60081 loops=1)"

"   Sort Key: js.""StudentAMKA"""

"   Sort Method: external merge Disk: 1536kB"

"   -> Nested Loop (cost=0.43..56711.71 rows=2698 width=16) (actual time=481.323..4545.198 rows=60081 loops=1)"

"   -> Nested Loop (cost=0.43..46016.22 rows=49142 width=16) (actual time=0.035..1586.547 rows=540089 loops=1)"

"   -> Seq Scan on ""Program"" pgr (cost=0.00..2.50 rows=1 width=4) (actual time=0.014..0.052 rows=11 loops=1)"

"   Filter: (""Duration"" = 5)"

"   Rows Removed by Filter: 29"

"   -> Index Only Scan using ""Joins_pkey"" on ""Joins"" js (cost=0.43..45522.30 rows=49142 width=16) (actual time=0.037..135.454 rows=49099 loops=11)"

"   Index Cond: (""ProgramID"" = pgr.""ProgramID"")"

"   Heap Fetches: 0"

"   -> Index Scan using idx_student_amka on ""Student"" (cost=0.00..0.22 rows=1 width=12) (actual time=0.005..0.005 rows=0 loops=540089)"

"   Index Cond: ((amka)::text = (js.""StudentAMKA"")::text)"

"   Rows Removed by Index Recheck: 0"

"   Filter: ((entry_date >= '2040-09-01'::date) AND (entry_date <= '2050-09-30'::date))"

"   Rows Removed by Filter: 1"

"   -> Index Only Scan using ""Student_pkey"" on ""Student"" s (cost=0.43..4.22 rows=1 width=12) (actual time=0.026..0.026 rows=1 loops=553)"

"   Index Cond: (amka = (js.""StudentAMKA"")::text)"

"   Heap Fetches: 0"

"   -> Index Scan using idx_person_amka on ""Person"" p (cost=0.00..0.25 rows=1 width=16) (actual time=0.027..0.027 rows=1 loops=553)"

"   Index Cond: ((amka)::text = (s.amka)::text)"

"Planning Time: 1.045 ms"

"Execution Time: 5221.472 ms"

(Person: amka | Student: amka | Program: Duration)

ΔΟΚΙΜΗ 1η

"Sort (cost=61025.82..61026.93 rows=445 width=30) (actual time=5361.168..5361.194 rows=445 loops=1)"

" Sort Key: (CASE WHEN (c.population > '50000'::numeric) THEN COALESCE(counts.total_students, '0'::bigint) ELSE '0'::bigint END) DESC"

" Sort Method: quicksort Memory: 61kB"

" -> Hash Left Join (cost=60992.50..61006.25 rows=445 width=30) (actual time=5360.817..5361.087 rows=445 loops=1)"

"   Hash Cond: (c.id = counts.city_id)"

"   -> Seq Scan on ""Cities"" c (cost=0.00..11.45 rows=445 width=31) (actual time=0.019..0.125 rows=445 loops=1)"

"   -> Hash (cost=60992.15..60992.15 rows=28 width=12) (actual time=5360.782..5360.787 rows=28 loops=1)"

"   Buckets: 1024 Batches: 1 Memory Usage: 10kB"

"   -> Subquery Scan on counts (cost=60984.85..60992.15 rows=28 width=12) (actual time=5360.111..5360.773 rows=28 loops=1)"

"   -> GroupAggregate (cost=60984.85..60991.87 rows=28 width=12) (actual time=5360.110..5360.767 rows=28 loops=1)"

"   Group Key: p.city_id"

"   -> Sort (cost=60984.85..60987.10 rows=899 width=16) (actual time=5357.810..5357.857 rows=553 loops=1)"

"   Sort Key: p.city_id"

"   Sort Method: quicksort Memory: 50kB"

"   -> Nested Loop (cost=56865.89..60940.75 rows=899 width=16) (actual time=5255.878..5357.481 rows=553 loops=1)"

"   -> Nested Loop (cost=56865.89..60720.47 rows=899 width=24) (actual time=5255.832..5343.307 rows=553 loops=1)"

"   -> GroupAggregate (cost=56865.47..56919.43 rows=899 width=12) (actual time=5255.753..5328.030 rows=553 loops=1)"

"   Group Key: js.""StudentAMKA"""

"   Filter: (count(js.""ProgramID"") >= 2)"

"   Rows Removed by Filter: 58972"

"   -> Sort (cost=56865.47..56872.21 rows=2698 width=16) (actual time=5255.715..5298.608 rows=60081 loops=1)"

"   Sort Key: js.""StudentAMKA"""

"   Sort Method: external merge Disk: 1536kB"

"   -> Nested Loop (cost=0.43..56711.71 rows=2698 width=16) (actual time=380.319..4638.639 rows=60081 loops=1)"

"   -> Nested Loop (cost=0.43..46016.22 rows=49142 width=16) (actual time=0.046..1523.061 rows=540089 loops=1)"

" -> Seq Scan on ""Program"" pgr (cost=0.00..2.50 rows=1 width=4) (actual time=0.009..0.076 rows=11 loops=1)"

" Filter: (""Duration"" = 5)"

" Rows Removed by Filter: 29"

" -> Index Only Scan using ""Joins_pkey"" on ""Joins"" js (cost=0.43..45522.30 rows=49142 width=16) (actual time=0.034..130.532 rows=49099 loops=11)"

" Index Cond: (""ProgramID"" = pgr.""ProgramID"")"

" Heap Fetches: 0"

" -> Index Scan using idx_student_amka on ""Student"" (cost=0.00..0.22 rows=1 width=12) (actual time=0.005..0.005 rows=0 loops=540089)"

" Index Cond: ((amka)::text = (js.""StudentAMKA"")::text)"

" Rows Removed by Index Recheck: 0"

" Filter: ((entry_date >= '2040-09-01'::date) AND (entry_date <= '2050-09-30'::date))"

" Rows Removed by Filter: 1"

" -> Index Only Scan using ""Student_pkey"" on ""Student"" s (cost=0.43..4.22 rows=1 width=12) (actual time=0.026..0.026 rows=1 loops=553)"

" Index Cond: (amka = (js.""StudentAMKA"")::text)"

" Heap Fetches: 0"

" -> Index Scan using idx_person_amka on ""Person"" p (cost=0.00..0.25 rows=1 width=16) (actual time=0.025..0.025 rows=1 loops=553)"

" Index Cond: ((amka)::text = (s.amka)::text)"

"Planning Time: 3.600 ms"

"Execution Time: 5362.692 ms"

ΔΟΚΙΜΉ 2Η

"Sort (cost=61025.82..61026.93 rows=445 width=30) (actual time=5527.132..5527.159 rows=445 loops=1)"

" Sort Key: (CASE WHEN (c.population > '50000'::numeric) THEN COALESCE(counts.total_students, '0'::bigint) ELSE '0'::bigint END) DESC"

" Sort Method: quicksort Memory: 61kB"

" -> Hash Left Join (cost=60992.50..61006.25 rows=445 width=30) (actual time=5526.817..5527.058 rows=445 loops=1)"

"   Hash Cond: (c.id = counts.city_id)"

"   -> Seq Scan on ""Cities"" c (cost=0.00..11.45 rows=445 width=31) (actual time=0.072..0.152 rows=445 loops=1)"

"   -> Hash (cost=60992.15..60992.15 rows=28 width=12) (actual time=5526.731..5526.734 rows=28 loops=1)"

"   Buckets: 1024 Batches: 1 Memory Usage: 10kB"

"   -> Subquery Scan on counts (cost=60984.85..60992.15 rows=28 width=12) (actual time=5526.315..5526.721 rows=28 loops=1)"

"   -> GroupAggregate (cost=60984.85..60991.87 rows=28 width=12) (actual time=5526.314..5526.716 rows=28 loops=1)"

"   Group Key: p.city_id"

"   -> Sort (cost=60984.85..60987.10 rows=899 width=16) (actual time=5524.521..5524.556 rows=553 loops=1)"

"   Sort Key: p.city_id"

"   Sort Method: quicksort Memory: 50kB"

"   -> Nested Loop (cost=56865.89..60940.75 rows=899 width=16) (actual time=5426.469..5524.175 rows=553 loops=1)"

"   -> Nested Loop (cost=56865.89..60720.47 rows=899 width=24) (actual time=5426.417..5508.583 rows=553 loops=1)"

"   -> GroupAggregate (cost=56865.47..56919.43 rows=899 width=12) (actual time=5426.305..5493.097 rows=553 loops=1)"

"   Group Key: js.""StudentAMKA"""

"   Filter: (count(js.""ProgramID"") >= 2)"

"   Rows Removed by Filter: 58972"

"   -> Sort (cost=56865.47..56872.21 rows=2698 width=16) (actual time=5426.267..5462.654 rows=60081 loops=1)"

"   Sort Key: js.""StudentAMKA"""

"   Sort Method: external merge Disk: 1536kB"

"   -> Nested Loop (cost=0.43..56711.71 rows=2698 width=16) (actual time=550.211..4753.834 rows=60081 loops=1)"

"   -> Nested Loop (cost=0.43..46016.22 rows=49142 width=16) (actual time=0.054..1608.304 rows=540089 loops=1)"

"   -> Seq Scan on ""Program"" pgr (cost=0.00..2.50 rows=1 width=4) (actual time=0.027..0.110 rows=11 loops=1)"

"   Filter: (""Duration"" = 5)"

"   Rows Removed by Filter: 29"

"   -> Index Only Scan using ""Joins_pkey"" on ""Joins"" js (cost=0.43..45522.30 rows=49142 width=16) (actual time=0.034..137.135 rows=49099 loops=11)"

"   Index Cond: (""ProgramID"" = pgr.""ProgramID"")"

"   Heap Fetches: 0"

"   -> Index Scan using idx_student_amka on ""Student"" (cost=0.00..0.22 rows=1 width=12) (actual time=0.005..0.005 rows=0 loops=540089)"

"   Index Cond: ((amka)::text = (js.""StudentAMKA"")::text)"

"   Rows Removed by Index Recheck: 0"

"   Filter: ((entry_date >= '2040-09-01'::date) AND (entry_date <= '2050-09-30'::date))"

"   Rows Removed by Filter: 1"

"   -> Index Only Scan using ""Student_pkey"" on ""Student"" s (cost=0.43..4.22 rows=1 width=12) (actual time=0.027..0.027 rows=1 loops=553)"

"   Index Cond: (amka = (js.""StudentAMKA"")::text)"

"   Heap Fetches: 0"

"   -> Index Scan using idx_person_amka on ""Person"" p (cost=0.00..0.25 rows=1 width=16) (actual time=0.027..0.027 rows=1 loops=553)"

"   Index Cond: ((amka)::text = (s.amka)::text)"

"Planning Time: 1.687 ms"

"Execution Time: 5528.644 ms"

ΔΟΚΙΜΉ 3Η

"Sort (cost=61025.82..61026.93 rows=445 width=30) (actual time=7282.556..7282.584 rows=445 loops=1)"

" Sort Key: (CASE WHEN (c.population > '50000'::numeric) THEN COALESCE(counts.total_students, '0'::bigint) ELSE '0'::bigint END) DESC"

" Sort Method: quicksort Memory: 61kB"

" -> Hash Left Join (cost=60992.50..61006.25 rows=445 width=30) (actual time=7282.250..7282.489 rows=445 loops=1)"

"   Hash Cond: (c.id = counts.city_id)"

"   -> Seq Scan on ""Cities"" c (cost=0.00..11.45 rows=445 width=31) (actual time=0.014..0.092 rows=445 loops=1)"

"   -> Hash (cost=60992.15..60992.15 rows=28 width=12) (actual time=7282.227..7282.232 rows=28 loops=1)"

"   Buckets: 1024 Batches: 1 Memory Usage: 10kB"

"   -> Subquery Scan on counts (cost=60984.85..60992.15 rows=28 width=12) (actual time=7281.842..7282.223 rows=28 loops=1)"

"   -> GroupAggregate (cost=60984.85..60991.87 rows=28 width=12) (actual time=7281.841..7282.218 rows=28 loops=1)"

"   Group Key: p.city_id"

"   -> Sort (cost=60984.85..60987.10 rows=899 width=16) (actual time=7280.107..7280.139 rows=553 loops=1)"

"   Sort Key: p.city_id"

"   Sort Method: quicksort Memory: 50kB"

"   -> Nested Loop (cost=56865.89..60940.75 rows=899 width=16) (actual time=7185.199..7279.643 rows=553 loops=1)"

"   -> Nested Loop (cost=56865.89..60720.47 rows=899 width=24) (actual time=7185.143..7264.691 rows=553 loops=1)"

"   -> GroupAggregate (cost=56865.47..56919.43 rows=899 width=12) (actual time=7185.051..7250.281 rows=553 loops=1)"

"   Group Key: js.""StudentAMKA"""

"   Filter: (count(js.""ProgramID"") >= 2)"

"   Rows Removed by Filter: 58972"

"   -> Sort (cost=56865.47..56872.21 rows=2698 width=16) (actual time=7185.011..7221.310 rows=60081 loops=1)"

"   Sort Key: js.""StudentAMKA"""

"   Sort Method: external merge Disk: 1536kB"

"   -> Nested Loop (cost=0.43..56711.71 rows=2698 width=16) (actual time=1469.957..6570.655 rows=60081 loops=1)"

"   -> Nested Loop (cost=0.43..46016.22 rows=49142 width=16) (actual time=0.033..1645.058 rows=540089 loops=1)"

" -> Seq Scan on ""Program"" pgr (cost=0.00..2.50 rows=1 width=4) (actual time=0.014..0.053 rows=11 loops=1)"

" Filter: (""Duration"" = 5)"

" Rows Removed by Filter: 29"

" -> Index Only Scan using ""Joins_pkey"" on ""Joins"" js (cost=0.43..45522.30 rows=49142 width=16) (actual time=0.034..140.953 rows=49099 loops=11)"

" Index Cond: (""ProgramID"" = pgr.""ProgramID"")"

" Heap Fetches: 0"

" -> Index Scan using idx_student_amka on ""Student"" (cost=0.00..0.22 rows=1 width=12) (actual time=0.009..0.009 rows=0 loops=540089)"

" Index Cond: ((amka)::text = (js.""StudentAMKA"")::text)"

" Rows Removed by Index Recheck: 0"

" Filter: ((entry_date >= '2040-09-01'::date) AND (entry_date <= '2050-09-30'::date))"

" Rows Removed by Filter: 1"

" -> Index Only Scan using ""Student_pkey"" on ""Student"" s (cost=0.43..4.22 rows=1 width=12) (actual time=0.025..0.025 rows=1 loops=553)"

" Index Cond: (amka = (js.""StudentAMKA"")::text)"

" Heap Fetches: 0"

" -> Index Scan using idx_person_amka on ""Person"" p (cost=0.00..0.25 rows=1 width=16) (actual time=0.026..0.026 rows=1 loops=553)"

" Index Cond: ((amka)::text = (s.amka)::text)"

"Planning Time: 0.986 ms"

"Execution Time: 7283.793 ms"

(Person: amka | Student: amka | Program: Duration, ProgramID)

ΔΟΚΙΜΗ 1η

"Sort (cost=61025.82..61026.93 rows=445 width=30) (actual time=5917.671..5917.712 rows=445 loops=1)"

" Sort Key: (CASE WHEN (c.population > '50000'::numeric) THEN COALESCE(counts.total_students, '0'::bigint) ELSE '0'::bigint END) DESC"

" Sort Method: quicksort Memory: 61kB"

" -> Hash Left Join (cost=60992.50..61006.25 rows=445 width=30) (actual time=5917.343..5917.601 rows=445 loops=1)"

"   Hash Cond: (c.id = counts.city_id)"

"   -> Seq Scan on ""Cities"" c (cost=0.00..11.45 rows=445 width=31) (actual time=0.018..0.111 rows=445 loops=1)"

"   -> Hash (cost=60992.15..60992.15 rows=28 width=12) (actual time=5917.316..5917.321 rows=28 loops=1)"

"   Buckets: 1024 Batches: 1 Memory Usage: 10kB"

"   -> Subquery Scan on counts (cost=60984.85..60992.15 rows=28 width=12) (actual time=5916.928..5917.312 rows=28 loops=1)"

"   -> GroupAggregate (cost=60984.85..60991.87 rows=28 width=12) (actual time=5916.928..5917.307 rows=28 loops=1)"

"   Group Key: p.city_id"

"   -> Sort (cost=60984.85..60987.10 rows=899 width=16) (actual time=5915.153..5915.187 rows=553 loops=1)"

"   Sort Key: p.city_id"

"   Sort Method: quicksort Memory: 50kB"

"   -> Nested Loop (cost=56865.89..60940.75 rows=899 width=16) (actual time=5800.479..5914.511 rows=553 loops=1)"

"   -> Nested Loop (cost=56865.89..60720.47 rows=899 width=24) (actual time=5800.433..5893.675 rows=553 loops=1)"

"   -> GroupAggregate (cost=56865.47..56919.43 rows=899 width=12) (actual time=5800.357..5875.962 rows=553 loops=1)"

"   Group Key: js.""StudentAMKA"""

"   Filter: (count(js.""ProgramID"") >= 2)"

"   Rows Removed by Filter: 58972"

"   -> Sort (cost=56865.47..56872.21 rows=2698 width=16) (actual time=5800.320..5839.049 rows=60081 loops=1)"

"   Sort Key: js.""StudentAMKA"""

"   Sort Method: external merge Disk: 1536kB"

"   -> Nested Loop (cost=0.43..56711.71 rows=2698 width=16) (actual time=384.820..5079.207 rows=60081 loops=1)"

"   -> Nested Loop (cost=0.43..46016.22 rows=49142 width=16) (actual time=0.031..1702.056 rows=540089 loops=1)"

" -> Seq Scan on ""Program"" pgr (cost=0.00..2.50 rows=1 width=4) (actual time=0.008..0.070 rows=11 loops=1)"

" Filter: (""Duration"" = 5)"

" Rows Removed by Filter: 29"

" -> Index Only Scan using ""Joins_pkey"" on ""Joins"" js (cost=0.43..45522.30 rows=49142 width=16) (actual time=0.043..146.238 rows=49099 loops=11)"

" Index Cond: (""ProgramID"" = pgr.""ProgramID"")"

" Heap Fetches: 0"

" -> Index Scan using idx_student_amka on ""Student"" (cost=0.00..0.22 rows=1 width=12) (actual time=0.006..0.006 rows=0 loops=540089)"

" Index Cond: ((amka)::text = (js.""StudentAMKA"")::text)"

" Rows Removed by Index Recheck: 0"

" Filter: ((entry_date >= '2040-09-01'::date) AND (entry_date <= '2050-09-30'::date))"

" Rows Removed by Filter: 1"

" -> Index Only Scan using ""Student_pkey"" on ""Student"" s (cost=0.43..4.22 rows=1 width=12) (actual time=0.030..0.030 rows=1 loops=553)"

" Index Cond: (amka = (js.""StudentAMKA"")::text)"

" Heap Fetches: 0"

" -> Index Scan using idx_person_amka on ""Person"" p (cost=0.00..0.25 rows=1 width=16) (actual time=0.036..0.036 rows=1 loops=553)"

" Index Cond: ((amka)::text = (s.amka)::text)"

"Planning Time: 2.493 ms"

"Execution Time: 5918.975 ms"

ΔΟΚΙΜΉΗ 2Η

"Sort (cost=61025.82..61026.93 rows=445 width=30) (actual time=5761.112..5761.145 rows=445 loops=1)"

" Sort Key: (CASE WHEN (c.population > '50000'::numeric) THEN COALESCE(counts.total_students, '0'::bigint) ELSE '0'::bigint END) DESC"

" Sort Method: quicksort Memory: 61kB"

" -> Hash Left Join (cost=60992.50..61006.25 rows=445 width=30) (actual time=5760.749..5761.043 rows=445 loops=1)"

"   Hash Cond: (c.id = counts.city_id)"

"   -> Seq Scan on ""Cities"" c (cost=0.00..11.45 rows=445 width=31) (actual time=0.021..0.150 rows=445 loops=1)"

"   -> Hash (cost=60992.15..60992.15 rows=28 width=12) (actual time=5760.717..5760.724 rows=28 loops=1)"

"   Buckets: 1024 Batches: 1 Memory Usage: 10kB"

"   -> Subquery Scan on counts (cost=60984.85..60992.15 rows=28 width=12) (actual time=5760.312..5760.711 rows=28 loops=1)"

"   -> GroupAggregate (cost=60984.85..60991.87 rows=28 width=12) (actual time=5760.311..5760.705 rows=28 loops=1)"

"   Group Key: p.city_id"

"   -> Sort (cost=60984.85..60987.10 rows=899 width=16) (actual time=5758.311..5758.355 rows=553 loops=1)"

"   Sort Key: p.city_id"

"   Sort Method: quicksort Memory: 50kB"

"   -> Nested Loop (cost=56865.89..60940.75 rows=899 width=16) (actual time=5662.263..5758.018 rows=553 loops=1)"

"   -> Nested Loop (cost=56865.89..60720.47 rows=899 width=24) (actual time=5662.188..5743.142 rows=553 loops=1)"

"   -> GroupAggregate (cost=56865.47..56919.43 rows=899 width=12) (actual time=5662.110..5729.287 rows=553 loops=1)"

"   Group Key: js.""StudentAMKA"""

"   Filter: (count(js.""ProgramID"") >= 2)"

"   Rows Removed by Filter: 58972"

"   -> Sort (cost=56865.47..56872.21 rows=2698 width=16) (actual time=5662.070..5699.346 rows=60081 loops=1)"

"   Sort Key: js.""StudentAMKA"""

"   Sort Method: external merge Disk: 1536kB"

"   -> Nested Loop (cost=0.43..56711.71 rows=2698 width=16) (actual time=502.931..5103.769 rows=60081 loops=1)"

"   -> Nested Loop (cost=0.43..46016.22 rows=49142 width=16) (actual time=0.042..1789.903 rows=540089 loops=1)"

" -> Seq Scan on ""Program"" pgr (cost=0.00..2.50 rows=1 width=4) (actual time=0.021..0.058 rows=11 loops=1)"

" Filter: (""Duration"" = 5)"

" Rows Removed by Filter: 29"

" -> Index Only Scan using ""Joins_pkey"" on ""Joins"" js (cost=0.43..45522.30 rows=49142 width=16) (actual time=0.035..154.099 rows=49099 loops=11)"

" Index Cond: (""ProgramID"" = pgr.""ProgramID"")"

" Heap Fetches: 0"

" -> Index Scan using idx_student_amka on ""Student"" (cost=0.00..0.22 rows=1 width=12) (actual time=0.006..0.006 rows=0 loops=540089)"

" Index Cond: ((amka)::text = (js.""StudentAMKA"")::text)"

" Rows Removed by Index Recheck: 0"

" Filter: ((entry_date >= '2040-09-01'::date) AND (entry_date <= '2050-09-30'::date))"

" Rows Removed by Filter: 1"

" -> Index Only Scan using ""Student_pkey"" on ""Student"" s (cost=0.43..4.22 rows=1 width=12) (actual time=0.024..0.024 rows=1 loops=553)"

" Index Cond: (amka = (js.""StudentAMKA"")::text)"

" Heap Fetches: 0"

" -> Index Scan using idx_person_amka on ""Person"" p (cost=0.00..0.25 rows=1 width=16) (actual time=0.026..0.026 rows=1 loops=553)"

" Index Cond: ((amka)::text = (s.amka)::text)"

"Planning Time: 1.679 ms"

"Execution Time: 5762.602 ms"

ΔΟΚΙΜΉ 3Η

"Sort (cost=61025.82..61026.93 rows=445 width=30) (actual time=8497.623..8497.664 rows=445 loops=1)"

" Sort Key: (CASE WHEN (c.population > '50000'::numeric) THEN COALESCE(counts.total_students, '0'::bigint) ELSE '0'::bigint END) DESC"

" Sort Method: quicksort Memory: 61kB"

" -> Hash Left Join (cost=60992.50..61006.25 rows=445 width=30) (actual time=8497.077..8497.539 rows=445 loops=1)"

"   Hash Cond: (c.id = counts.city_id)"

"   -> Seq Scan on ""Cities"" c (cost=0.00..11.45 rows=445 width=31) (actual time=0.023..0.287 rows=445 loops=1)"

"   -> Hash (cost=60992.15..60992.15 rows=28 width=12) (actual time=8497.036..8497.049 rows=28 loops=1)"

"   Buckets: 1024 Batches: 1 Memory Usage: 10kB"

"   -> Subquery Scan on counts (cost=60984.85..60992.15 rows=28 width=12) (actual time=8496.619..8497.032 rows=28 loops=1)"

"   -> GroupAggregate (cost=60984.85..60991.87 rows=28 width=12) (actual time=8496.617..8497.025 rows=28 loops=1)"

"   Group Key: p.city_id"

"   -> Sort (cost=60984.85..60987.10 rows=899 width=16) (actual time=8494.741..8494.790 rows=553 loops=1)"

"   Sort Key: p.city_id"

"   Sort Method: quicksort Memory: 50kB"

"   -> Nested Loop (cost=56865.89..60940.75 rows=899 width=16) (actual time=8398.249..8494.386 rows=553 loops=1)"

"   -> Nested Loop (cost=56865.89..60720.47 rows=899 width=24) (actual time=8398.204..8479.445 rows=553 loops=1)"

"   -> GroupAggregate (cost=56865.47..56919.43 rows=899 width=12) (actual time=8398.065..8464.884 rows=553 loops=1)"

"   Group Key: js.""StudentAMKA"""

"   Filter: (count(js.""ProgramID"") >= 2)"

"   Rows Removed by Filter: 58972"

"   -> Sort (cost=56865.47..56872.21 rows=2698 width=16) (actual time=8398.005..8434.765 rows=60081 loops=1)"

"   Sort Key: js.""StudentAMKA"""

"   Sort Method: external merge Disk: 1536kB"

"   -> Nested Loop (cost=0.43..56711.71 rows=2698 width=16) (actual time=1626.228..7793.968 rows=60081 loops=1)"

"   -> Nested Loop (cost=0.43..46016.22 rows=49142 width=16) (actual time=0.055..1605.625 rows=540089 loops=1)"

" -> Seq Scan on ""Program"" pgr (cost=0.00..2.50 rows=1 width=4) (actual time=0.019..0.053 rows=11 loops=1)"

" Filter: (""Duration"" = 5)"

" Rows Removed by Filter: 29"

" -> Index Only Scan using ""Joins_pkey"" on ""Joins"" js (cost=0.43..45522.30 rows=49142 width=16) (actual time=0.047..137.317 rows=49099 loops=11)"

" Index Cond: (""ProgramID"" = pgr.""ProgramID"")"

" Heap Fetches: 0"

" -> Index Scan using idx_student_amka on ""Student"" (cost=0.00..0.22 rows=1 width=12) (actual time=0.011..0.011 rows=0 loops=540089)"

" Index Cond: ((amka)::text = (js.""StudentAMKA"")::text)"

" Rows Removed by Index Recheck: 0"

" Filter: ((entry_date >= '2040-09-01'::date) AND (entry_date <= '2050-09-30'::date))"

" Rows Removed by Filter: 1"

" -> Index Only Scan using ""Student_pkey"" on ""Student"" s (cost=0.43..4.22 rows=1 width=12) (actual time=0.025..0.025 rows=1 loops=553)"

" Index Cond: (amka = (js.""StudentAMKA"")::text)"

" Heap Fetches: 0"

" -> Index Scan using idx_person_amka on ""Person"" p (cost=0.00..0.25 rows=1 width=16) (actual time=0.026..0.026 rows=1 loops=553)"

" Index Cond: ((amka)::text = (s.amka)::text)"

"Planning Time: 1.031 ms"

"Execution Time: 8499.421 ms"

Query με χρήση συνδυασμού B-Tree και Hash βέλτιστων ευρετηρίων

(B-Tree Student:entry_date | Hash Program: Duration)

ΔΟΚΙΜΉ 1Η

"Sort (cost=59069.55..59070.66 rows=445 width=30) (actual time=2401.145..2401.177 rows=445 loops=1)"

" Sort Key: (CASE WHEN (c.population > '50000'::numeric) THEN COALESCE(counts.total_students, '0'::bigint) ELSE '0'::bigint END) DESC"

" Sort Method: quicksort Memory: 61kB"

" -> Hash Left Join (cost=59036.23..59049.98 rows=445 width=30) (actual time=2400.361..2400.990 rows=445 loops=1)"

"   Hash Cond: (c.id = counts.city_id)"

"   -> Seq Scan on ""Cities"" c (cost=0.00..11.45 rows=445 width=31) (actual time=0.020..0.281 rows=445 loops=1)"

"   -> Hash (cost=59035.88..59035.88 rows=28 width=12) (actual time=2400.271..2400.280 rows=28 loops=1)"

"   Buckets: 1024 Batches: 1 Memory Usage: 10kB"

"   -> Subquery Scan on counts (cost=59028.58..59035.88 rows=28 width=12) (actual time=2399.865..2400.267 rows=28 loops=1)"

"   -> GroupAggregate (cost=59028.58..59035.60 rows=28 width=12) (actual time=2399.864..2400.261 rows=28 loops=1)"

"   Group Key: p.city_id"

"   -> Sort (cost=59028.58..59030.83 rows=899 width=16) (actual time=2397.220..2397.264 rows=553 loops=1)"

"   Sort Key: p.city_id"

"   Sort Method: quicksort Memory: 50kB"

"   -> Nested Loop (cost=54527.15..58984.48 rows=899 width=16) (actual time=2206.393..2395.749 rows=553 loops=1)"

"   -> Nested Loop (cost=54526.72..58381.30 rows=899 width=24) (actual time=2205.314..2281.822 rows=553 loops=1)"

"   -> GroupAggregate (cost=54526.29..54580.25 rows=899 width=12) (actual time=2205.265..2271.518 rows=553 loops=1)"

"   Group Key: js.""StudentAMKA"""

"   Filter: (count(js.""ProgramID"") >= 2)"

"   Rows Removed by Filter: 58972"

"   -> Sort (cost=54526.29..54533.04 rows=2698 width=16) (actual time=2205.226..2241.686 rows=60081 loops=1)"

"   Sort Key: js.""StudentAMKA"""

"   Sort Method: external merge Disk: 1536kB"

"   -> Hash Join (cost=46630.93..54372.54 rows=2698 width=16) (actual time=1090.726..1601.670 rows=60081 loops=1)"

" Hash Cond: (((""Student"".amka)::text = (js.""StudentAMKA"")::text)"

" -> Index Scan using idx_student_entry_date on ""Student"" (cost=0.43..7317.85 rows=105922 width=12) (actual time=0.057..75.812 rows=108262 loops=1)"

" Index Cond: ((entry_date >= '2040-09-01'::date) AND (entry_date <= '2050-09-30'::date))"

" -> Hash (cost=46016.22..46016.22 rows=49142 width=16) (actual time=1088.107..1088.109 rows=540089 loops=1)"

" Buckets: 131072 (originally 65536) Batches: 16 (originally 1) Memory Usage: 3073kB"

" -> Nested Loop (cost=0.43..46016.22 rows=49142 width=16) (actual time=0.027..853.049 rows=540089 loops=1)"

" -> Seq Scan on ""Program"" pgr (cost=0.00..2.50 rows=1 width=4) (actual time=0.011..0.054 rows=11 loops=1)"

" Filter: (""Duration"" = 5)"

" Rows Removed by Filter: 29"

" -> Index Only Scan using ""Joins_pkey"" on ""Joins"" js (cost=0.43..45522.30 rows=49142 width=16) (actual time=0.012..70.083 rows=49099 loops=11)"

" Index Cond: (""ProgramID"" = pgr.""ProgramID"")"

" Heap Fetches: 0"

" -> Index Only Scan using ""Student_pkey"" on ""Student"" s (cost=0.43..4.22 rows=1 width=12) (actual time=0.016..0.016 rows=1 loops=553)"

" Index Cond: (amka = (js.""StudentAMKA"")::text)"

" Heap Fetches: 0"

" -> Index Scan using ""Person_pkey"" on ""Person"" p (cost=0.43..0.67 rows=1 width=16) (actual time=0.203..0.203 rows=1 loops=553)"

" Index Cond: ((amka)::text = (s.amka)::text)"

"Planning Time: 5.642 ms"

"Execution Time: 2404.431 ms"

ΔΟΚΙΜΉ 2Η

"Sort (cost=59069.55..59070.66 rows=445 width=30) (actual time=2474.078..2474.106 rows=445 loops=1)"

" Sort Key: (CASE WHEN (c.population > '50000'::numeric) THEN COALESCE(counts.total_students, '0'::bigint) ELSE '0'::bigint END) DESC"

" Sort Method: quicksort Memory: 61kB"

" -> Hash Left Join (cost=59036.23..59049.98 rows=445 width=30) (actual time=2473.794..2474.006 rows=445 loops=1)"

"   Hash Cond: (c.id = counts.city_id)"

"   -> Seq Scan on ""Cities"" c (cost=0.00..11.45 rows=445 width=31) (actual time=0.015..0.067 rows=445 loops=1)"

"   -> Hash (cost=59035.88..59035.88 rows=28 width=12) (actual time=2473.768..2473.772 rows=28 loops=1)"

"   Buckets: 1024 Batches: 1 Memory Usage: 10kB"

"   -> Subquery Scan on counts (cost=59028.58..59035.88 rows=28 width=12) (actual time=2473.389..2473.762 rows=28 loops=1)"

"   -> GroupAggregate (cost=59028.58..59035.60 rows=28 width=12) (actual time=2473.389..2473.758 rows=28 loops=1)"

"   Group Key: p.city_id"

"   -> Sort (cost=59028.58..59030.83 rows=899 width=16) (actual time=2471.640..2471.672 rows=553 loops=1)"

"   Sort Key: p.city_id"

"   Sort Method: quicksort Memory: 50kB"

"   -> Nested Loop (cost=54527.15..58984.48 rows=899 width=16) (actual time=2392.557..2471.451 rows=553 loops=1)"

"   -> Nested Loop (cost=54526.72..58381.30 rows=899 width=24) (actual time=2392.514..2463.543 rows=553 loops=1)"

"   -> GroupAggregate (cost=54526.29..54580.25 rows=899 width=12) (actual time=2392.302..2455.542 rows=553 loops=1)"

"   Group Key: js.""StudentAMKA"""

"   Filter: (count(js.""ProgramID"") >= 2)"

"   Rows Removed by Filter: 58972"

"   -> Sort (cost=54526.29..54533.04 rows=2698 width=16) (actual time=2392.252..2426.635 rows=60081 loops=1)"

"   Sort Key: js.""StudentAMKA"""

"   Sort Method: external merge Disk: 1536kB"

"   -> Hash Join (cost=46630.93..54372.54 rows=2698 width=16) (actual time=1259.277..1878.844 rows=60081 loops=1)"

"   Hash Cond: (((""Student"".amka)::text = (js.""StudentAMKA"")::text)"

"   -> Index Scan using idx_student_entry_date on ""Student"" (cost=0.43..7317.85 rows=105922 width=12) (actual time=0.018..70.792 rows=108262 loops=1)"

"   Index Cond: ((entry_date >= '2040-09-01'::date) AND (entry_date <= '2050-09-30'::date))"

"   -> Hash (cost=46016.22..46016.22 rows=49142 width=16) (actual time=1257.256..1257.258 rows=540089 loops=1)"

"   Buckets: 131072 (originally 65536) Batches: 16 (originally 1) Memory Usage: 3073kB"

"   -> Nested Loop (cost=0.43..46016.22 rows=49142 width=16) (actual time=0.026..968.439 rows=540089 loops=1)"

"   -> Seq Scan on ""Program"" pgr (cost=0.00..2.50 rows=1 width=4) (actual time=0.011..0.077 rows=11 loops=1)"

"   Filter: (""Duration"" = 5)"

"   Rows Removed by Filter: 29"

"   -> Index Only Scan using ""Joins_pkey"" on ""Joins"" js (cost=0.43..45522.30 rows=49142 width=16) (actual time=0.016..79.433 rows=49099 loops=11)"

"   Index Cond: (""ProgramID"" = pgr.""ProgramID"")"

"   Heap Fetches: 0"

"   -> Index Only Scan using ""Student_pkey"" on ""Student"" s (cost=0.43..4.22 rows=1 width=12) (actual time=0.014..0.014 rows=1 loops=553)"

"   Index Cond: (amka = (js.""StudentAMKA"")::text)"

"   Heap Fetches: 0"

"   -> Index Scan using ""Person_pkey"" on ""Person"" p (cost=0.43..0.67 rows=1 width=16) (actual time=0.013..0.013 rows=1 loops=553)"

"   Index Cond: ((amka)::text = (s.amka)::text)"

"Planning Time: 0.732 ms"

"Execution Time: 2478.299 ms"

ΔΟΚΙΜΉ 3Η

"Sort (cost=59069.55..59070.66 rows=445 width=30) (actual time=2557.705..2557.734 rows=445 loops=1)"

" Sort Key: (CASE WHEN (c.population > '50000'::numeric) THEN COALESCE(counts.total_students, '0'::bigint) ELSE '0'::bigint END) DESC"

" Sort Method: quicksort Memory: 61kB"

" -> Hash Left Join (cost=59036.23..59049.98 rows=445 width=30) (actual time=2557.421..2557.633 rows=445 loops=1)"

"   Hash Cond: (c.id = counts.city_id)"

"   -> Seq Scan on ""Cities"" c (cost=0.00..11.45 rows=445 width=31) (actual time=0.019..0.071 rows=445 loops=1)"

"   -> Hash (cost=59035.88..59035.88 rows=28 width=12) (actual time=2557.391..2557.396 rows=28 loops=1)"

"   Buckets: 1024 Batches: 1 Memory Usage: 10kB"

"   -> Subquery Scan on counts (cost=59028.58..59035.88 rows=28 width=12) (actual time=2557.009..2557.386 rows=28 loops=1)"

"   -> GroupAggregate (cost=59028.58..59035.60 rows=28 width=12) (actual time=2557.009..2557.381 rows=28 loops=1)"

"   Group Key: p.city_id"

"   -> Sort (cost=59028.58..59030.83 rows=899 width=16) (actual time=2555.245..2555.279 rows=553 loops=1)"

"   Sort Key: p.city_id"

"   Sort Method: quicksort Memory: 50kB"

"   -> Nested Loop (cost=54527.15..58984.48 rows=899 width=16) (actual time=2473.195..2555.040 rows=553 loops=1)"

"   -> Nested Loop (cost=54526.72..58381.30 rows=899 width=24) (actual time=2473.151..2546.819 rows=553 loops=1)"

"   -> GroupAggregate (cost=54526.29..54580.25 rows=899 width=12) (actual time=2473.062..2539.019 rows=553 loops=1)"

"   Group Key: js.""StudentAMKA"""

"   Filter: (count(js.""ProgramID"") >= 2)"

"   Rows Removed by Filter: 58972"

"   -> Sort (cost=54526.29..54533.04 rows=2698 width=16) (actual time=2473.004..2508.707 rows=60081 loops=1)"

"   Sort Key: js.""StudentAMKA"""

"   Sort Method: external merge Disk: 1536kB"

"   -> Hash Join (cost=46630.93..54372.54 rows=2698 width=16) (actual time=1481.635..1976.440 rows=60081 loops=1)"

"   Hash Cond: ((""Student"".amka)::text = (js.""StudentAMKA"")::text)"

"   -> Index Scan using idx_student_entry_date on ""Student"" (cost=0.43..7317.85 rows=105922 width=12) (actual time=0.017..101.790 rows=108262 loops=1)"

" Index Cond: ((entry_date >= '2040-09-01'::date) AND (entry_date <= '2050-09-30'::date))"

" -> Hash (cost=46016.22..46016.22 rows=49142 width=16) (actual time=1478.855..1478.857 rows=540089 loops=1)"

" Buckets: 131072 (originally 65536) Batches: 16 (originally 1) Memory Usage: 3073kB"

" -> Nested Loop (cost=0.43..46016.22 rows=49142 width=16) (actual time=0.025..1132.273 rows=540089 loops=1)"

" -> Seq Scan on ""Program"" pgr (cost=0.00..2.50 rows=1 width=4) (actual time=0.011..0.066 rows=11 loops=1)"

" Filter: (""Duration"" = 5)"

" Rows Removed by Filter: 29"

" -> Index Only Scan using ""Joins_pkey"" on ""Joins"" js (cost=0.43..45522.30 rows=49142 width=16) (actual time=0.015..93.233 rows=49099 loops=11)"

" Index Cond: (""ProgramID"" = pgr.""ProgramID"")"

" Heap Fetches: 0"

" -> Index Only Scan using ""Student_pkey"" on ""Student"" s (cost=0.43..4.22 rows=1 width=12) (actual time=0.013..0.013 rows=1 loops=553)"

" Index Cond: (amka = (js.""StudentAMKA"")::text)"

" Heap Fetches: 0"

" -> Index Scan using ""Person_pkey"" on ""Person"" p (cost=0.43..0.67 rows=1 width=16) (actual time=0.014..0.014 rows=1 loops=553)"

" Index Cond: ((amka)::text = (s.amka)::text)"

"Planning Time: 0.726 ms"

"Execution Time: 2562.927 ms"

Query με χρήση συνδυασμού B-Tree και Hash βέλτιστων ευρετηρίων με χρήση cluster των αντίστοιχων ευρετηρίων.

(B-Tree Student:entry_date | Hash Program: Duration)

ΔΟΚΙΜΉ 1Η

"Sort (cost=59069.55..59070.66 rows=445 width=30) (actual time=2422.024..2422.054 rows=445 loops=1)"

" Sort Key: (CASE WHEN (c.population > '50000'::numeric) THEN COALESCE(counts.total_students, '0'::bigint) ELSE '0'::bigint END) DESC"

" Sort Method: quicksort Memory: 61kB"

" -> Hash Left Join (cost=59036.23..59049.98 rows=445 width=30) (actual time=2421.737..2421.943 rows=445 loops=1)"

"   Hash Cond: (c.id = counts.city_id)"

"   -> Seq Scan on ""Cities"" c (cost=0.00..11.45 rows=445 width=31) (actual time=0.025..0.074 rows=445 loops=1)"

"   -> Hash (cost=59035.88..59035.88 rows=28 width=12) (actual time=2421.702..2421.706 rows=28 loops=1)"

"   Buckets: 1024 Batches: 1 Memory Usage: 10kB"

"   -> Subquery Scan on counts (cost=59028.58..59035.88 rows=28 width=12) (actual time=2421.322..2421.696 rows=28 loops=1)"

"   -> GroupAggregate (cost=59028.58..59035.60 rows=28 width=12) (actual time=2421.321..2421.690 rows=28 loops=1)"

"   Group Key: p.city_id"

"   -> Sort (cost=59028.58..59030.83 rows=899 width=16) (actual time=2419.576..2419.608 rows=553 loops=1)"

"   Sort Key: p.city_id"

"   Sort Method: quicksort Memory: 50kB"

"   -> Nested Loop (cost=54527.15..58984.48 rows=899 width=16) (actual time=2336.991..2419.375 rows=553 loops=1)"

"   -> Nested Loop (cost=54526.72..58381.30 rows=899 width=24) (actual time=2336.966..2411.559 rows=553 loops=1)"

"   -> GroupAggregate (cost=54526.29..54580.25 rows=899 width=12) (actual time=2336.918..2401.137 rows=553 loops=1)"

"   Group Key: js.""StudentAMKA"""

"   Filter: (count(js.""ProgramID"") >= 2)"

"   Rows Removed by Filter: 58972"

"   -> Sort (cost=54526.29..54533.04 rows=2698 width=16) (actual time=2336.879..2370.900 rows=60081 loops=1)"

"   Sort Key: js.""StudentAMKA"""

"   Sort Method: external merge Disk: 1536kB"

" -> Hash Join (cost=46630.93..54372.54 rows=2698 width=16) (actual time=1225.580..1726.572 rows=60081 loops=1)"

" Hash Cond: (((""Student"".amka)::text = (js.""StudentAMKA"")::text)"

" -> Index Scan using idx_student_entry_date on ""Student"" (cost=0.43..7317.85 rows=105922 width=12) (actual time=0.018..75.290 rows=108262 loops=1)"

" Index Cond: ((entry_date >= '2040-09-01'::date) AND (entry_date <= '2050-09-30'::date))"

" -> Hash (cost=46016.22..46016.22 rows=49142 width=16) (actual time=1222.664..1222.666 rows=540089 loops=1)"

" Buckets: 131072 (originally 65536) Batches: 16 (originally 1) Memory Usage: 3073kB"

" -> Nested Loop (cost=0.43..46016.22 rows=49142 width=16) (actual time=0.031..942.171 rows=540089 loops=1)"

" -> Seq Scan on ""Program"" pgr (cost=0.00..2.50 rows=1 width=4) (actual time=0.012..0.321 rows=11 loops=1)"

" Filter: (""Duration"" = 5)"

" Rows Removed by Filter: 29"

" -> Index Only Scan using ""Joins_pkey"" on ""Joins"" js (cost=0.43..45522.30 rows=49142 width=16) (actual time=0.017..77.063 rows=49099 loops=11)"

" Index Cond: (""ProgramID"" = pgr.""ProgramID"")"

" Heap Fetches: 0"

" -> Index Only Scan using ""Student_pkey"" on ""Student"" s (cost=0.43..4.22 rows=1 width=12) (actual time=0.018..0.018 rows=1 loops=553)"

" Index Cond: (amka = (js.""StudentAMKA"")::text)"

" Heap Fetches: 0"

" -> Index Scan using ""Person_pkey"" on ""Person"" p (cost=0.43..0.67 rows=1 width=16) (actual time=0.013..0.013 rows=1 loops=553)"

" Index Cond: ((amka)::text = (s.amka)::text)"

"Planning Time: 0.895 ms"

"Execution Time: 2426.799 ms"

ΔΟΚΙΜΉ 2Η

"Sort (cost=41402.38..41403.49 rows=445 width=30) (actual time=1131.036..1132.176 rows=445 loops=1)"

" Sort Key: (CASE WHEN (c.population > '50000'::numeric) THEN COALESCE(counts.total_students, '0'::bigint) ELSE '0'::bigint END) DESC"

" Sort Method: quicksort Memory: 61kB"

" -> Hash Left Join (cost=41369.06..41382.81 rows=445 width=30) (actual time=1130.721..1132.056 rows=445 loops=1)"

"   Hash Cond: (c.id = counts.city_id)"

"   -> Seq Scan on ""Cities"" c (cost=0.00..11.45 rows=445 width=31) (actual time=0.011..0.075 rows=445 loops=1)"

"   -> Hash (cost=41368.04..41368.04 rows=82 width=12) (actual time=1130.700..1131.812 rows=28 loops=1)"

"   Buckets: 1024 Batches: 1 Memory Usage: 10kB"

"   -> Subquery Scan on counts (cost=41365.78..41368.04 rows=82 width=12) (actual time=1130.255..1131.796 rows=28 loops=1)"

"   -> GroupAggregate (cost=41365.78..41367.22 rows=82 width=12) (actual time=1130.254..1131.790 rows=28 loops=1)"

"   Group Key: p.city_id"

"   -> Sort (cost=41365.78..41365.99 rows=82 width=36) (actual time=1127.436..1128.610 rows=553 loops=1)"

"   Sort Key: p.city_id"

"   Sort Method: quicksort Memory: 50kB"

"   -> Nested Loop (cost=40588.63..41363.18 rows=82 width=36) (actual time=971.703..1128.252 rows=553 loops=1)"

"   -> Nested Loop (cost=40588.21..41310.51 rows=82 width=48) (actual time=965.890..1113.367 rows=553 loops=1)"

"   -> Finalize GroupAggregate (cost=40587.78..40617.20 rows=82 width=12) (actual time=965.842..1101.232 rows=553 loops=1)"

"   Group Key: js.""StudentAMKA"""

"   Filter: (count(js.""ProgramID"") >= 2)"

"   Rows Removed by Filter: 58972"

"   -> Gather Merge (cost=40587.78..40613.11 rows=204 width=20) (actual time=965.792..1071.659 rows=59525 loops=1)"

"   Workers Planned: 2"

"   Workers Launched: 2"

"   -> Partial GroupAggregate (cost=39587.75..39589.54 rows=102 width=20) (actual time=857.565..873.452 rows=19842 loops=3)"

"   Group Key: js.""StudentAMKA"""

"   -> Sort (cost=39587.75..39588.01 rows=102 width=16) (actual time=857.552..861.926 rows=20027 loops=3)"

"   Sort Key: js.""StudentAMKA"""

"   Sort Method: quicksort Memory: 1947kB"

"   Worker 0: Sort Method: quicksort Memory: 1672kB"

"   Worker 1: Sort Method: quicksort Memory: 1118kB"

" -> Hash Join (cost=138.25..39584.35 rows=102 width=16) (actual time=3.196..699.632 rows=20027 loops=3)"

" Hash Cond: (js.""ProgramID"" = pgr.""ProgramID"")"

" -> Nested Loop (cost=135.74..39569.95 rows=4095 width=16) (actual time=2.113..680.372 rows=37177 loops=3)"

" -> Parallel Bitmap Heap Scan on ""Student"" (cost=135.31..27916.50 rows=4020 width=32) (actual time=1.568..18.369 rows=36087 loops=3)"

" Recheck Cond: ((entry_date >= '2040-09-01'::date) AND (entry_date <= '2050-09-30'::date))"

" Heap Blocks: exact=1781"

" -> Bitmap Index Scan on idx_student_entry_date (cost=0.00..132.90 rows=9647 width=0) (actual time=3.755..3.756 rows=108262 loops=1)"

" Index Cond: ((entry_date >= '2040-09-01'::date) AND (entry_date <= '2050-09-30'::date))"

" -> Index Only Scan using ""Joins_pkey"" on ""Joins"" js (cost=0.43..2.89 rows=1 width=16) (actual time=0.017..0.018 rows=1 loops=108262)"

" Index Cond: (""StudentAMKA"" = (""Student"".amka)::text)"

" Heap Fetches: 0"

" -> Hash (cost=2.50..2.50 rows=1 width=4) (actual time=0.383..0.385 rows=11 loops=3)"

" Buckets: 1024 Batches: 1 Memory Usage: 9kB"

" -> Seq Scan on ""Program"" pgr (cost=0.00..2.50 rows=1 width=4) (actual time=0.360..0.369 rows=11 loops=3)"

" Filter: (""Duration"" = 5)"

" Rows Removed by Filter: 29"

" -> Index Scan using ""Person_pkey"" on ""Person"" p (cost=0.43..8.45 rows=1 width=36) (actual time=0.018..0.018 rows=1 loops=553)"

" Index Cond: ((amka)::text = (js.""StudentAMKA"")::text)"

" -> Index Only Scan using ""Student_pkey"" on ""Student"" s (cost=0.43..0.64 rows=1 width=32) (actual time=0.026..0.026 rows=1 loops=553)"

" Index Cond: (amka = (p.amka)::text)"

" Heap Fetches: 553"

"Planning Time: 4.290 ms"

"Execution Time: 1132.712 ms"

ΔΟΚΙΜΉ 3Η

"Sort (cost=41402.38..41403.49 rows=445 width=30) (actual time=1688.724..1688.984 rows=445 loops=1)"

" Sort Key: (CASE WHEN (c.population > '50000'::numeric) THEN COALESCE(counts.total_students, '0'::bigint) ELSE '0'::bigint END) DESC"

" Sort Method: quicksort Memory: 61kB"

" -> Hash Left Join (cost=41369.06..41382.81 rows=445 width=30) (actual time=1688.338..1688.864 rows=445 loops=1)"

"   Hash Cond: (c.id = counts.city_id)"

"   -> Seq Scan on ""Cities"" c (cost=0.00..11.45 rows=445 width=31) (actual time=0.013..0.074 rows=445 loops=1)"

"   -> Hash (cost=41368.04..41368.04 rows=82 width=12) (actual time=1688.314..1688.549 rows=28 loops=1)"

"   Buckets: 1024 Batches: 1 Memory Usage: 10kB"

"   -> Subquery Scan on counts (cost=41365.78..41368.04 rows=82 width=12) (actual time=1687.835..1688.534 rows=28 loops=1)"

"   -> GroupAggregate (cost=41365.78..41367.22 rows=82 width=12) (actual time=1687.834..1688.527 rows=28 loops=1)"

"   Group Key: p.city_id"

"   -> Sort (cost=41365.78..41365.99 rows=82 width=36) (actual time=1677.518..1677.791 rows=553 loops=1)"

"   Sort Key: p.city_id"

"   Sort Method: quicksort Memory: 50kB"

"   -> Nested Loop (cost=40588.63..41363.18 rows=82 width=36) (actual time=1457.446..1677.474 rows=553 loops=1)"

"   -> Nested Loop (cost=40588.21..41310.51 rows=82 width=48) (actual time=1457.417..1668.025 rows=553 loops=1)"

"   -> Finalize GroupAggregate (cost=40587.78..40617.20 rows=82 width=12) (actual time=1457.348..1656.690 rows=553 loops=1)"

"   Group Key: js.""StudentAMKA"""

"   Filter: (count(js.""ProgramID"") >= 2)"

"   Rows Removed by Filter: 58972"

"   -> Gather Merge (cost=40587.78..40613.11 rows=204 width=20) (actual time=1457.287..1627.359 rows=59525 loops=1)"

"   Workers Planned: 2"

"   Workers Launched: 2"

"   -> Partial GroupAggregate (cost=39587.75..39589.54 rows=102 width=20) (actual time=1278.805..1294.876 rows=19842 loops=3)"

"   Group Key: js.""StudentAMKA"""

"   -> Sort (cost=39587.75..39588.01 rows=102 width=16) (actual time=1278.786..1283.096 rows=20027 loops=3)"

"   Sort Key: js.""StudentAMKA"""

"   Sort Method: quicksort Memory: 1956kB"

"   Worker 0: Sort Method: quicksort Memory: 1553kB"

"   Worker 1: Sort Method: quicksort Memory: 1613kB"

" -> Hash Join (cost=138.25..39584.35 rows=102 width=16) (actual time=3.265..948.145 rows=20027 loops=3)"

" Hash Cond: (js.""ProgramID"" = pgr.""ProgramID"")"

" -> Nested Loop (cost=135.74..39569.95 rows=4095 width=16) (actual time=1.658..916.163 rows=37177 loops=3)"

" -> Parallel Bitmap Heap Scan on ""Student"" (cost=135.31..27916.50 rows=4020 width=32) (actual time=1.287..15.085 rows=36087 loops=3)"

" Recheck Cond: ((entry_date >= '2040-09-01'::date) AND (entry_date <= '2050-09-30'::date))"

" Heap Blocks: exact=1868"

" -> Bitmap Index Scan on idx_student_entry_date (cost=0.00..132.90 rows=9647 width=0) (actual time=2.890..2.890 rows=108262 loops=1)"

" Index Cond: ((entry_date >= '2040-09-01'::date) AND (entry_date <= '2050-09-30'::date))"

" -> Index Only Scan using ""Joins_pkey"" on ""Joins"" js (cost=0.43..2.89 rows=1 width=16) (actual time=0.024..0.024 rows=1 loops=108262)"

" Index Cond: (""StudentAMKA"" = (""Student"".amka)::text)"

" Heap Fetches: 0"

" -> Hash (cost=2.50..2.50 rows=1 width=4) (actual time=1.566..1.567 rows=11 loops=3)"

" Buckets: 1024 Batches: 1 Memory Usage: 9kB"

" -> Seq Scan on ""Program"" pgr (cost=0.00..2.50 rows=1 width=4) (actual time=1.541..1.550 rows=11 loops=3)"

" Filter: (""Duration"" = 5)"

" Rows Removed by Filter: 29"

" -> Index Scan using ""Person_pkey"" on ""Person"" p (cost=0.43..8.45 rows=1 width=36) (actual time=0.019..0.019 rows=1 loops=553)"

" Index Cond: ((amka)::text = (js.""StudentAMKA"")::text)"

" -> Index Only Scan using ""Student_pkey"" on ""Student"" s (cost=0.43..0.64 rows=1 width=32) (actual time=0.016..0.016 rows=1 loops=553)"

" Index Cond: (amka = (p.amka)::text)"

" Heap Fetches: 0"

"Planning Time: 1.512 ms"

"Execution Time: 1689.479 ms"

Query με αλλαγμένη σειρά join , έπειτα από εισαγωγή δεδομένων και χρήση βέλτιστων ευρετηρίων μαζί με τα αντίστοιχα clusters.

HASH: Program: Duration | B-Tree: Student: entry_date | clusters και των δύο

ΔΟΚΙΜΉ 1Η

"Sort (cost=57234.25..57235.37 rows=445 width=30) (actual time=2667.630..2672.262 rows=445 loops=1)"

" Sort Key: (CASE WHEN (c.population > '50000'::numeric) THEN COALESCE(counts.total_students, '0'::bigint) ELSE '0'::bigint END) DESC"

" Sort Method: quicksort Memory: 61kB"

" -> Hash Left Join (cost=57200.94..57214.68 rows=445 width=30) (actual time=2666.820..2672.132 rows=445 loops=1)"

"   Hash Cond: (c.id = counts.city_id)"

"   -> Seq Scan on ""Cities"" c (cost=0.00..11.45 rows=445 width=31) (actual time=0.014..0.152 rows=445 loops=1)"

"   -> Hash (cost=57200.59..57200.59 rows=28 width=12) (actual time=2666.795..2671.390 rows=28 loops=1)"

"   Buckets: 1024 Batches: 1 Memory Usage: 10kB"

"   -> Subquery Scan on counts (cost=57193.59..57200.59 rows=28 width=12) (actual time=2666.341..2671.346 rows=28 loops=1)"

"   -> GroupAggregate (cost=57193.59..57200.31 rows=28 width=12) (actual time=2666.339..2671.339 rows=28 loops=1)"

"   Group Key: p.city_id"

"   -> Sort (cost=57193.59..57195.74 rows=858 width=16) (actual time=2664.337..2668.972 rows=553 loops=1)"

"   Sort Key: p.city_id"

"   Sort Method: quicksort Memory: 50kB"

"   -> Nested Loop (cost=52691.14..57151.79 rows=858 width=16) (actual time=2390.653..2668.397 rows=553 loops=1)"

"   Join Filter: ((js.""StudentAMKA"")::text = (p.amka)::text)"

"   -> Nested Loop (cost=52690.72..56565.39 rows=858 width=24) (actual time=2390.626..2654.420 rows=553 loops=1)"

"   -> Finalize GroupAggregate (cost=52690.29..52927.00 rows=858 width=12) (actual time=2390.565..2641.137 rows=553 loops=1)"

"   Group Key: js.""StudentAMKA"""

"   Filter: (count(js.""ProgramID"") >= 2)"

"   Rows Removed by Filter: 58972"

"   -> Gather Merge (cost=52690.29..52887.24 rows=1515 width=20) (actual time=2390.186..2596.478 rows=59525 loops=1)"

"   Workers Planned: 1"

"   Workers Launched: 1"

"   -> Partial GroupAggregate (cost=51690.28..51716.79 rows=1515 width=20) (actual time=2249.632..2284.693 rows=29763 loops=2)"

"   Group Key: js.""StudentAMKA"""

"   -> Sort (cost=51690.28..51694.07 rows=1515 width=16) (actual time=2249.469..2263.982 rows=30041 loops=2)"

" Sort Key: js.""StudentAMKA"""

" Sort Method: quicksort Memory: 2240kB"

" Worker 0: Sort Method: quicksort Memory: 2113kB"

" -> Hash Join (cost=3.37..51610.25 rows=1515 width=16) (actual time=0.744..1462.348 rows=30041 loops=2)"

" Hash Cond: (js.""ProgramID"" = pgr.""ProgramID"")"

" -> Nested Loop (cost=0.85..51431.84 rows=60586 width=16) (actual time=0.383..1433.672 rows=55766 loops=2)"

" -> Parallel Index Scan using idx_student_entry_date on ""Student"" (cost=0.43..6557.10 rows=59469 width=12) (actual time=0.025..42.531 rows=54131 loops=2)"

" Index Cond: ((entry_date >= '2040-09-01'::date) AND (entry_date <= '2050-09-30'::date))"

" -> Index Only Scan using ""Joins_pkey"" on ""Joins"" js (cost=0.43..0.74 rows=1 width=16) (actual time=0.024..0.025 rows=1 loops=108262)"

" Index Cond: (""StudentAMKA"" = (""Student"".amka)::text)"

" Heap Fetches: 0"

" -> Hash (cost=2.50..2.50 rows=1 width=4) (actual time=0.287..0.288 rows=11 loops=2)"

" Buckets: 1024 Batches: 1 Memory Usage: 9kB"

" -> Seq Scan on ""Program"" pgr (cost=0.00..2.50 rows=1 width=4) (actual time=0.267..0.276 rows=11 loops=2)"

" Filter: (""Duration"" = 5)"

" Rows Removed by Filter: 29"

" -> Index Only Scan using ""Student_pkey"" on ""Student"" s (cost=0.43..4.23 rows=1 width=12) (actual time=0.021..0.021 rows=1 loops=553)"

" Index Cond: (amka = (js.""StudentAMKA"")::text)"

" Heap Fetches: 0"

" -> Index Scan using ""Person_pkey"" on ""Person"" p (cost=0.43..0.67 rows=1 width=16) (actual time=0.022..0.022 rows=1 loops=553)"

" Index Cond: ((amka)::text = (s.amka)::text)"

"Planning Time: 11.104 ms"

"Execution Time: 2673.015 ms"

ΔΟΚΙΜΉ 2Η

"Sort (cost=57234.25..57235.37 rows=445 width=30) (actual time=1933.009..1936.876 rows=445 loops=1)"

" Sort Key: (CASE WHEN (c.population > '50000'::numeric) THEN COALESCE(counts.total_students, '0'::bigint) ELSE '0'::bigint END) DESC"

" Sort Method: quicksort Memory: 61kB"

" -> Hash Left Join (cost=57200.94..57214.68 rows=445 width=30) (actual time=1932.716..1936.774 rows=445 loops=1)"

"   Hash Cond: (c.id = counts.city_id)"

"   -> Seq Scan on ""Cities"" c (cost=0.00..11.45 rows=445 width=31) (actual time=0.013..0.071 rows=445 loops=1)"

"   -> Hash (cost=57200.59..57200.59 rows=28 width=12) (actual time=1932.694..1936.536 rows=28 loops=1)"

"   Buckets: 1024 Batches: 1 Memory Usage: 10kB"

"   -> Subquery Scan on counts (cost=57193.59..57200.59 rows=28 width=12) (actual time=1932.235..1936.516 rows=28 loops=1)"

"   -> GroupAggregate (cost=57193.59..57200.31 rows=28 width=12) (actual time=1932.234..1936.509 rows=28 loops=1)"

"   Group Key: p.city_id"

"   -> Sort (cost=57193.59..57195.74 rows=858 width=16) (actual time=1929.638..1933.516 rows=553 loops=1)"

"   Sort Key: p.city_id"

"   Sort Method: quicksort Memory: 50kB"

"   -> Nested Loop (cost=52691.14..57151.79 rows=858 width=16) (actual time=1785.883..1933.176 rows=553 loops=1)"

"   Join Filter: ((js.""StudentAMKA"")::text = (p.amka)::text)"

"   -> Nested Loop (cost=52690.72..56565.39 rows=858 width=24) (actual time=1785.848..1923.830 rows=553 loops=1)"

"   -> Finalize GroupAggregate (cost=52690.29..52927.00 rows=858 width=12) (actual time=1785.427..1913.317 rows=553 loops=1)"

"   Group Key: js.""StudentAMKA"""

"   Filter: (count(js.""ProgramID"") >= 2)"

"   Rows Removed by Filter: 58972"

"   -> Gather Merge (cost=52690.29..52887.24 rows=1515 width=20) (actual time=1785.330..1870.207 rows=59525 loops=1)"

"   Workers Planned: 1"

"   Workers Launched: 1"

"   -> Partial GroupAggregate (cost=51690.28..51716.79 rows=1515 width=20) (actual time=1743.469..1770.190 rows=29763 loops=2)"

"   Group Key: js.""StudentAMKA"""

" -> Sort (cost=51690.28..51694.07 rows=1515 width=16) (actual time=1743.454..1750.454 rows=30041 loops=2)"

" Sort Key: js.""StudentAMKA"""

" Sort Method: quicksort Memory: 2243kB"

" Worker 0: Sort Method: quicksort Memory: 2110kB"

" -> Hash Join (cost=3.37..51610.25 rows=1515 width=16) (actual time=0.898..1405.783 rows=30041 loops=2)"

" Hash Cond: (js.""ProgramID"" = pgr.""ProgramID"")"

" -> Nested Loop (cost=0.85..51431.84 rows=60586 width=16) (actual time=0.492..1384.971 rows=55766 loops=2)"

" -> Parallel Index Scan using idx_student_entry_date on ""Student"" (cost=0.43..6557.10 rows=59469 width=12) (actual time=0.021..27.276 rows=54131 loops=2)"

" Index Cond: ((entry_date >= '2040-09-01'::date) AND (entry_date <= '2050-09-30'::date))"

" -> Index Only Scan using ""Joins_pkey"" on ""Joins"" js (cost=0.43..0.74 rows=1 width=16) (actual time=0.024..0.024 rows=1 loops=108262)"

" Index Cond: (""StudentAMKA"" = (""Student"".amka)::text)"

" Heap Fetches: 0"

" -> Hash (cost=2.50..2.50 rows=1 width=4) (actual time=0.352..0.352 rows=11 loops=2)"

" Buckets: 1024 Batches: 1 Memory Usage: 9kB"

" -> Seq Scan on ""Program"" pgr (cost=0.00..2.50 rows=1 width=4) (actual time=0.329..0.339 rows=11 loops=2)"

" Filter: (""Duration"" = 5)"

" Rows Removed by Filter: 29"

" -> Index Only Scan using ""Student_pkey"" on ""Student"" s (cost=0.43..4.23 rows=1 width=12) (actual time=0.018..0.018 rows=1 loops=553)"

" Index Cond: (amka = (js.""StudentAMKA"")::text)"

" Heap Fetches: 0"

" -> Index Scan using ""Person_pkey"" on ""Person"" p (cost=0.43..0.67 rows=1 width=16) (actual time=0.015..0.015 rows=1 loops=553)"

" Index Cond: ((amka)::text = (s.amka)::text)"

"Planning Time: 0.742 ms"

"Execution Time: 1937.341 ms"

ΔΟΚΙΜΉ 3Η

"Sort (cost=57234.25..57235.37 rows=445 width=30) (actual time=1452.698..1456.176 rows=445 loops=1)"

" Sort Key: (CASE WHEN (c.population > '50000'::numeric) THEN COALESCE(counts.total_students, '0'::bigint) ELSE '0'::bigint END) DESC"

" Sort Method: quicksort Memory: 61kB"

" -> Hash Left Join (cost=57200.94..57214.68 rows=445 width=30) (actual time=1452.380..1456.065 rows=445 loops=1)"

"   Hash Cond: (c.id = counts.city_id)"

"   -> Seq Scan on ""Cities"" c (cost=0.00..11.45 rows=445 width=31) (actual time=0.026..0.084 rows=445 loops=1)"

"   -> Hash (cost=57200.59..57200.59 rows=28 width=12) (actual time=1452.344..1455.796 rows=28 loops=1)"

"   Buckets: 1024 Batches: 1 Memory Usage: 10kB"

"   -> Subquery Scan on counts (cost=57193.59..57200.59 rows=28 width=12) (actual time=1451.956..1455.783 rows=28 loops=1)"

"   -> GroupAggregate (cost=57193.59..57200.31 rows=28 width=12) (actual time=1451.955..1455.777 rows=28 loops=1)"

"   Group Key: p.city_id"

"   -> Sort (cost=57193.59..57195.74 rows=858 width=16) (actual time=1450.189..1453.668 rows=553 loops=1)"

"   Sort Key: p.city_id"

"   Sort Method: quicksort Memory: 50kB"

"   -> Nested Loop (cost=52691.14..57151.79 rows=858 width=16) (actual time=1326.667..1453.337 rows=553 loops=1)"

"   Join Filter: ((js.""StudentAMKA"")::text = (p.amka)::text)"

"   -> Nested Loop (cost=52690.72..56565.39 rows=858 width=24) (actual time=1326.640..1443.665 rows=553 loops=1)"

"   -> Finalize GroupAggregate (cost=52690.29..52927.00 rows=858 width=12) (actual time=1326.582..1433.984 rows=553 loops=1)"

"   Group Key: js.""StudentAMKA"""

"   Filter: (count(js.""ProgramID"") >= 2)"

"   Rows Removed by Filter: 58972"

"   -> Gather Merge (cost=52690.29..52887.24 rows=1515 width=20) (actual time=1326.540..1403.715 rows=59525 loops=1)"

"   Workers Planned: 1"

"   Workers Launched: 1"

"   -> Partial GroupAggregate (cost=51690.28..51716.79 rows=1515 width=20) (actual time=1262.776..1287.638 rows=29763 loops=2)"

"   Group Key: js.""StudentAMKA"""

"   -> Sort (cost=51690.28..51694.07 rows=1515 width=16) (actual time=1262.762..1269.704 rows=30041 loops=2)"

"   Sort Key: js.""StudentAMKA"""

"   Sort Method: quicksort Memory: 2925kB"

" Worker 0: Sort Method: quicksort Memory: 2026kB"

" -> Hash Join (cost=3.37..51610.25 rows=1515 width=16) (actual time=0.813..1008.220 rows=30041 loops=2)"

" Hash Cond: (js.""ProgramID"" = pgr.""ProgramID"")"

" -> Nested Loop (cost=0.85..51431.84 rows=60586 width=16) (actual time=0.352..990.070 rows=55766 loops=2)"

" -> Parallel Index Scan using idx_student_entry_date on ""Student"" (cost=0.43..6557.10 rows=59469 width=12) (actual time=0.025..24.780 rows=54131 loops=2)"

" Index Cond: ((entry_date >= '2040-09-01'::date) AND (entry_date <= '2050-09-30'::date))"

" -> Index Only Scan using ""Joins_pkey"" on ""Joins"" js (cost=0.43..0.74 rows=1 width=16) (actual time=0.017..0.017 rows=1 loops=108262)"

" Index Cond: (""StudentAMKA"" = (""Student"".amka)::text)"

" Heap Fetches: 0"

" -> Hash (cost=2.50..2.50 rows=1 width=4) (actual time=0.418..0.419 rows=11 loops=2)"

" Buckets: 1024 Batches: 1 Memory Usage: 9kB"

" -> Seq Scan on ""Program"" pgr (cost=0.00..2.50 rows=1 width=4) (actual time=0.396..0.405 rows=11 loops=2)"

" Filter: (""Duration"" = 5)"

" Rows Removed by Filter: 29"

" -> Index Only Scan using ""Student_pkey"" on ""Student"" s (cost=0.43..4.23 rows=1 width=12) (actual time=0.016..0.016 rows=1 loops=553)"

" Index Cond: (amka = (js.""StudentAMKA"")::text)"

" Heap Fetches: 0"

" -> Index Scan using ""Person_pkey"" on ""Person"" p (cost=0.43..0.67 rows=1 width=16) (actual time=0.016..0.016 rows=1 loops=553)"

" Index Cond: ((amka)::text = (s.amka)::text)"

"Planning Time: 0.761 ms"

"Execution Time: 1456.922 ms"

Query έπειτα από εισαγωγή δεδομένων στους πίνακες Person, Student, Joins, Program μέσω των συναρτήσεων που δημιουργήθηκαν, χωρίς ευρετήρια ή clustering.

(Οι μετρήσεις γίνανε σε καινούργιο database στο οποίο έγινε restored ξανά το backup αρχείο που δόθηκε, καθώς οι αλλαγές της συνάρτησης cluster δεν μπορούν να αναιρεθούν στην postgresql.)

ΔΟΚΙΜΉ 1Η

"Sort (cost=93576.27..93577.38 rows=445 width=30) (actual time=10887.805..10892.685 rows=445 loops=1)"

" Sort Key: (CASE WHEN (c.population > '50000'::numeric) THEN COALESCE(counts.total_students, '0'::bigint) ELSE '0'::bigint END) DESC"

" Sort Method: quicksort Memory: 61kB"

" -> Hash Left Join (cost=93542.95..93556.70 rows=445 width=30) (actual time=10885.550..10892.555 rows=445 loops=1)"

"   Hash Cond: (c.id = counts.city_id)"

"   -> Seq Scan on ""Cities"" c (cost=0.00..11.45 rows=445 width=31) (actual time=0.016..0.823 rows=445 loops=1)"

"   -> Hash (cost=93542.93..93542.93 rows=2 width=12) (actual time=10885.516..10890.366 rows=28 loops=1)"

"   Buckets: 1024 Batches: 1 Memory Usage: 10kB"

"   -> Subquery Scan on counts (cost=93542.87..93542.93 rows=2 width=12) (actual time=10885.135..10890.354 rows=28 loops=1)"

"   -> GroupAggregate (cost=93542.87..93542.91 rows=2 width=12) (actual time=10885.135..10890.348 rows=28 loops=1)"

"   Group Key: p.city_id"

"   -> Sort (cost=93542.87..93542.88 rows=2 width=16) (actual time=10883.347..10888.223 rows=553 loops=1)"

"   Sort Key: p.city_id"

"   Sort Method: quicksort Memory: 50kB"

"   -> Nested Loop (cost=93528.90..93542.86 rows=2 width=16) (actual time=10726.218..10887.786 rows=553 loops=1)"

"   -> Nested Loop (cost=93528.47..93541.55 rows=2 width=24) (actual time=10726.152..10865.557 rows=553 loops=1)"

"   -> Finalize GroupAggregate (cost=93528.04..93528.63 rows=2 width=12) (actual time=10725.999..10848.314 rows=553 loops=1)"

"   Group Key: js.""StudentAMKA"""

"   Filter: (count(js.""ProgramID"") >= 2)"

"   Rows Removed by Filter: 58972"

"   -> Gather Merge (cost=93528.04..93528.54 rows=4 width=20) (actual time=10725.955..10819.123 rows=59528 loops=1)"

"   Workers Planned: 2"

"   Workers Launched: 2"

" -> Partial GroupAggregate (cost=92528.02..92528.05 rows=2 width=20) (actual time=10623.374..10637.158 rows=19843 loops=3)"

" Group Key: js.""StudentAMKA"""

" -> Sort (cost=92528.02..92528.02 rows=2 width=16) (actual time=10623.360..10624.789 rows=20027 loops=3)"

" Sort Key: js.""StudentAMKA"""

" Sort Method: quicksort Memory: 1720kB"

" Worker 0: Sort Method: quicksort Memory: 1709kB"

" Worker 1: Sort Method: quicksort Memory: 1693kB"

" -> Nested Loop (cost=8780.41..92528.01 rows=2 width=16) (actual time=394.867..10580.120 rows=20027 loops=3)"

" -> Parallel Hash Join (cost=8779.98..92485.91 rows=68 width=16) (actual time=393.419..2320.387 rows=180030 loops=3)"

" Hash Cond: (js.""ProgramID"" = pgr.""ProgramID"")"

" -> Parallel Index Only Scan using ""Joins_pkey"" on ""Joins"" js (cost=0.43..81021.51 rows=1022792 width=16) (actual time=3.044..1667.364 rows=821887 loops=3)"

" Heap Fetches: 500334"

" -> Parallel Hash (cost=8779.38..8779.38 rows=14 width=4) (actual time=387.395..387.396 rows=4 loops=3)"

" Buckets: 1024 Batches: 1 Memory Usage: 40kB"

" -> Parallel Seq Scan on ""Program"" pgr (cost=0.00..8779.38 rows=14 width=4) (actual time=223.820..386.955 rows=4 loops=3)"

" Filter: (""Duration"" = 5)"

" Rows Removed by Filter: 166676"

" -> Index Scan using ""Student_pkey"" on ""Student"" (cost=0.43..0.62 rows=1 width=12) (actual time=0.045..0.045 rows=0 loops=540089)"

" Index Cond: ((amka)::text = (js.""StudentAMKA"")::text)"

" Filter: ((entry_date >= '2040-09-01'::date) AND (entry_date <= '2050-09-30'::date))"

" Rows Removed by Filter: 1"

" -> Index Only Scan using ""Student_pkey"" on ""Student"" s (cost=0.43..6.45 rows=1 width=12) (actual time=0.030..0.030 rows=1 loops=553)"

" Index Cond: (amka = (js.""StudentAMKA"")::text)"

" Heap Fetches: 0"

" -> Index Scan using ""Person_pkey"" on ""Person"" p (cost=0.43..0.66 rows=1 width=16) (actual time=0.038..0.038 rows=1 loops=553)"

" Index Cond: ((amka)::text = (s.amka)::text)"

"Planning Time: 8.674 ms" "Execution Time: 10893.961 ms"

ΔΟΚΙΜΉ 2Η

"Sort (cost=93576.27..93577.38 rows=445 width=30) (actual time=12823.973..12824.218 rows=445 loops=1)"

" Sort Key: (CASE WHEN (c.population > '50000'::numeric) THEN COALESCE(counts.total_students, '0'::bigint) ELSE '0'::bigint END) DESC"

" Sort Method: quicksort Memory: 61kB"

" -> Hash Left Join (cost=93542.95..93556.70 rows=445 width=30) (actual time=12823.610..12824.105 rows=445 loops=1)"

"   Hash Cond: (c.id = counts.city_id)"

"   -> Seq Scan on ""Cities"" c (cost=0.00..11.45 rows=445 width=31) (actual time=0.017..0.134 rows=445 loops=1)"

"   -> Hash (cost=93542.93..93542.93 rows=2 width=12) (actual time=12823.576..12823.795 rows=28 loops=1)"

"   Buckets: 1024 Batches: 1 Memory Usage: 10kB"

"   -> Subquery Scan on counts (cost=93542.87..93542.93 rows=2 width=12) (actual time=12823.070..12823.775 rows=28 loops=1)"

"   -> GroupAggregate (cost=93542.87..93542.91 rows=2 width=12) (actual time=12823.068..12823.767 rows=28 loops=1)"

"   Group Key: p.city_id"

"   -> Sort (cost=93542.87..93542.88 rows=2 width=16) (actual time=12820.947..12821.270 rows=553 loops=1)"

"   Sort Key: p.city_id"

"   Sort Method: quicksort Memory: 50kB"

"   -> Nested Loop (cost=93528.90..93542.86 rows=2 width=16) (actual time=12189.597..12819.503 rows=553 loops=1)"

"   -> Nested Loop (cost=93528.47..93541.55 rows=2 width=24) (actual time=12187.753..12442.027 rows=553 loops=1)"

"   -> Finalize GroupAggregate (cost=93528.04..93528.63 rows=2 width=12) (actual time=12185.886..12301.061 rows=553 loops=1)"

"   Group Key: js.""StudentAMKA"""

"   Filter: (count(js.""ProgramID"") >= 2)"

"   Rows Removed by Filter: 58972"

"   -> Gather Merge (cost=93528.04..93528.54 rows=4 width=20) (actual time=12185.831..12271.742 rows=59528 loops=1)"

"   Workers Planned: 2"

"   Workers Launched: 2"

"   -> Partial GroupAggregate (cost=92528.02..92528.05 rows=2 width=20) (actual time=12116.331..12129.445 rows=19843 loops=3)"

"   Group Key: js.""StudentAMKA"""

"   -> Sort (cost=92528.02..92528.02 rows=2 width=16) (actual time=12116.318..12117.702 rows=20027 loops=3)"

"   Sort Key: js.""StudentAMKA"""

"   Sort Method: quicksort Memory: 1714kB"

"   Worker 0: Sort Method: quicksort Memory: 1714kB"

"   Worker 1: Sort Method: quicksort Memory: 1693kB"

"   -> Nested Loop (cost=8780.41..92528.01 rows=2 width=16) (actual time=474.513..12074.897 rows=20027 loops=3)"

"   -> Parallel Hash Join (cost=8779.98..92485.91 rows=68 width=16) (actual time=472.412..2729.920 rows=180030 loops=3)"

"   Hash Cond: (js.""ProgramID"" = pgr.""ProgramID"")"

"   -> Parallel Index Only Scan using ""Joins_pkey"" on ""Joins"" js (cost=0.43..81021.51 rows=1022792 width=16) (actual time=3.452..2023.755 rows=821887 loops=3)"

"   Heap Fetches: 500334"

"   -> Parallel Hash (cost=8779.38..8779.38 rows=14 width=4) (actual time=468.774..468.774 rows=4 loops=3)"

"   Buckets: 1024 Batches: 1 Memory Usage: 40kB"

"   -> Parallel Seq Scan on ""Program"" pgr (cost=0.00..8779.38 rows=14 width=4) (actual time=289.941..468.684 rows=4 loops=3)"

"   Filter: (""Duration"" = 5)"

"   Rows Removed by Filter: 166676"

"   -> Index Scan using ""Student_pkey"" on ""Student"" (cost=0.43..0.62 rows=1 width=12) (actual time=0.051..0.051 rows=0 loops=540089)"

"   Index Cond: ((amka)::text = (js.""StudentAMKA"")::text)"

"   Filter: ((entry_date >= '2040-09-01'::date) AND (entry_date <= '2050-09-30'::date))"

"   Rows Removed by Filter: 1"

"   -> Index Only Scan using ""Student_pkey"" on ""Student"" s (cost=0.43..6.45 rows=1 width=12) (actual time=0.250..0.250 rows=1 loops=553)"

"   Index Cond: (amka = (js.""StudentAMKA"")::text)"

"   Heap Fetches: 0"

"   -> Index Scan using ""Person_pkey"" on ""Person"" p (cost=0.43..0.66 rows=1 width=16) (actual time=0.677..0.677 rows=1 loops=553)"

"   Index Cond: ((amka)::text = (s.amka)::text)"

"Planning Time: 8.348 ms"

"Execution Time: 12824.564 ms"

ΔΟΚΙΜΉΗ 3Η

"Sort (cost=93576.27..93577.38 rows=445 width=30) (actual time=10557.912..10558.205 rows=445 loops=1)"

" Sort Key: (CASE WHEN (c.population > '50000'::numeric) THEN COALESCE(counts.total_students, '0'::bigint) ELSE '0'::bigint END) DESC"

" Sort Method: quicksort Memory: 61kB"

" -> Hash Left Join (cost=93542.95..93556.70 rows=445 width=30) (actual time=10557.592..10558.101 rows=445 loops=1)"

"   Hash Cond: (c.id = counts.city_id)"

"   -> Seq Scan on ""Cities"" c (cost=0.00..11.45 rows=445 width=31) (actual time=0.019..0.102 rows=445 loops=1)"

"   -> Hash (cost=93542.93..93542.93 rows=2 width=12) (actual time=10557.562..10557.830 rows=28 loops=1)"

"   Buckets: 1024 Batches: 1 Memory Usage: 10kB"

"   -> Subquery Scan on counts (cost=93542.87..93542.93 rows=2 width=12) (actual time=10557.180..10557.819 rows=28 loops=1)"

"   -> GroupAggregate (cost=93542.87..93542.91 rows=2 width=12) (actual time=10557.179..10557.814 rows=28 loops=1)"

"   Group Key: p.city_id"

"   -> Sort (cost=93542.87..93542.88 rows=2 width=16) (actual time=10555.428..10555.726 rows=553 loops=1)"

"   Sort Key: p.city_id"

"   Sort Method: quicksort Memory: 50kB"

"   -> Nested Loop (cost=93528.90..93542.86 rows=2 width=16) (actual time=9941.817..10554.038 rows=553 loops=1)"

"   -> Nested Loop (cost=93528.47..93541.55 rows=2 width=24) (actual time=9940.429..10185.406 rows=553 loops=1)"

"   -> Finalize GroupAggregate (cost=93528.04..93528.63 rows=2 width=12) (actual time=9938.108..10051.359 rows=553 loops=1)"

"   Group Key: js.""StudentAMKA"""

"   Filter: (count(js.""ProgramID"") >= 2)"

"   Rows Removed by Filter: 58972"

"   -> Gather Merge (cost=93528.04..93528.54 rows=4 width=20) (actual time=9938.061..10021.972 rows=59528 loops=1)"

"   Workers Planned: 2"

"   Workers Launched: 2"

"   -> Partial GroupAggregate (cost=92528.02..92528.05 rows=2 width=20) (actual time=9875.880..9888.715 rows=19843 loops=3)"

"   Group Key: js.""StudentAMKA"""

" -> Sort (cost=92528.02..92528.02 rows=2 width=16) (actual time=9875.870..9877.357 rows=20027 loops=3)"

" Sort Key: js.""StudentAMKA"""

" Sort Method: quicksort Memory: 1708kB"

" Worker 0: Sort Method: quicksort Memory: 1707kB"

" Worker 1: Sort Method: quicksort Memory: 1707kB"

" -> Nested Loop (cost=8780.41..92528.01 rows=2 width=16) (actual time=430.006..9839.753 rows=20027 loops=3)"

" -> Parallel Hash Join (cost=8779.98..92485.91 rows=68 width=16) (actual time=428.427..2689.663 rows=180030 loops=3)"

" Hash Cond: (js.""ProgramID"" = pgr.""ProgramID"")"

" -> Parallel Index Only Scan using ""Joins_pkey"" on ""Joins"" js (cost=0.43..81021.51 rows=1022792 width=16) (actual time=1.973..2042.017 rows=821887 loops=3)"

" Heap Fetches: 500334"

" -> Parallel Hash (cost=8779.38..8779.38 rows=14 width=4) (actual time=424.822..424.823 rows=4 loops=3)"

" Buckets: 1024 Batches: 1 Memory Usage: 40kB"

" -> Parallel Seq Scan on ""Program"" pgr (cost=0.00..8779.38 rows=14 width=4) (actual time=262.256..424.748 rows=4 loops=3)"

" Filter: (""Duration"" = 5)"

" Rows Removed by Filter: 166676"

" -> Index Scan using ""Student_pkey"" on ""Student"" (cost=0.43..0.62 rows=1 width=12) (actual time=0.039..0.039 rows=0 loops=540089)"

" Index Cond: ((amka)::text = (js.""StudentAMKA"")::text)"

" Filter: ((entry_date >= '2040-09-01'::date) AND (entry_date <= '2050-09-30'::date))"

" Rows Removed by Filter: 1"

" -> Index Only Scan using ""Student_pkey"" on ""Student"" s (cost=0.43..6.45 rows=1 width=12) (actual time=0.238..0.238 rows=1 loops=553)"

" Index Cond: (amka = (js.""StudentAMKA"")::text)"

" Heap Fetches: 0"

" -> Index Scan using ""Person_pkey"" on ""Person"" p (cost=0.43..0.66 rows=1 width=16) (actual time=0.661..0.661 rows=1 loops=553)"

" Index Cond: ((amka)::text = (s.amka)::text)"

"Planning Time: 8.350 ms"

"Execution Time: 10558.576 ms"

Query έπειτα από εισαγωγή δεδομένων και χρήση ευρετηρίων B-Tree:

Student: entry_date | Program: Duration

ΔΟΚΙΜΉ 1Η

"Sort (cost=84806.24..84807.35 rows=445 width=30) (actual time=7301.820..7308.217 rows=445 loops=1)"

" Sort Key: (CASE WHEN (c.population > '50000'::numeric) THEN COALESCE(counts.total_students, '0'::bigint) ELSE '0'::bigint END) DESC"

" Sort Method: quicksort Memory: 61kB"

" -> Hash Left Join (cost=84772.92..84786.66 rows=445 width=30) (actual time=7301.475..7308.101 rows=445 loops=1)"

"   Hash Cond: (c.id = counts.city_id)"

"   -> Seq Scan on ""Cities"" c (cost=0.00..11.45 rows=445 width=31) (actual time=0.013..0.110 rows=445 loops=1)"

"   -> Hash (cost=84772.90..84772.90 rows=2 width=12) (actual time=7301.452..7307.823 rows=28 loops=1)"

"   Buckets: 1024 Batches: 1 Memory Usage: 10kB"

"   -> Subquery Scan on counts (cost=84772.84..84772.90 rows=2 width=12) (actual time=7301.032..7307.812 rows=28 loops=1)"

"   -> GroupAggregate (cost=84772.84..84772.88 rows=2 width=12) (actual time=7301.031..7307.807 rows=28 loops=1)"

"   Group Key: p.city_id"

"   -> Sort (cost=84772.84..84772.85 rows=2 width=16) (actual time=7299.213..7305.613 rows=553 loops=1)"

"   Sort Key: p.city_id"

"   Sort Method: quicksort Memory: 50kB"

"   -> Nested Loop (cost=84758.87..84772.83 rows=2 width=16) (actual time=7129.114..7305.178 rows=553 loops=1)"

"   -> Nested Loop (cost=84758.44..84771.51 rows=2 width=24) (actual time=7129.057..7284.437 rows=553 loops=1)"

"   -> Finalize GroupAggregate (cost=84758.01..84758.60 rows=2 width=12) (actual time=7128.958..7266.307 rows=553 loops=1)"

"   Group Key: js.""StudentAMKA"""

"   Filter: (count(js.""ProgramID"") >= 2)"

"   Rows Removed by Filter: 58972"

"   -> Gather Merge (cost=84758.01..84758.50 rows=4 width=20) (actual time=7128.895..7236.853 rows=59528 loops=1)"

"   Workers Planned: 2"

"   Workers Launched: 2"

"   -> Partial GroupAggregate (cost=83757.98..83758.02 rows=2 width=20) (actual time=7049.533..7063.341 rows=19843 loops=3)"

"   Group Key: js.""StudentAMKA"""

"   -> Sort (cost=83757.98..83757.99 rows=2 width=16) (actual time=7049.521..7050.854 rows=20027 loops=3)"

"   Sort Key: js.""StudentAMKA"""

"   Sort Method: quicksort Memory: 1722kB"

"   Worker 0: Sort Method: quicksort Memory: 1701kB"

"   Worker 1: Sort Method: quicksort Memory: 1699kB"

"   -> Nested Loop (cost=10.37..83757.97 rows=2 width=16) (actual time=10.424..7013.381 rows=20027 loops=3)"

"   -> Hash Join (cost=9.94..83715.88 rows=68 width=16) (actual time=8.101..982.803 rows=180030 loops=3)"

"   Hash Cond: (js.""ProgramID"" = pgr.""ProgramID"")"

"   -> Parallel Index Only Scan using ""Joins_pkey"" on ""Joins"" js (cost=0.43..81021.51 rows=1022792 width=16) (actual time=6.439..782.689 rows=821887 loops=3)"

"   Heap Fetches: 500334"

"   -> Hash (cost=9.10..9.10 rows=33 width=4) (actual time=0.646..0.647 rows=11 loops=3)"

"   Buckets: 1024 Batches: 1 Memory Usage: 9kB"

"   -> Index Scan using idx_program_duration on ""Program"" pgr (cost=0.42..9.10 rows=33 width=4) (actual time=0.613..0.623 rows=11 loops=3)"

"   Index Cond: (""Duration"" = 5)"

"   -> Index Scan using ""Student_pkey"" on ""Student"" (cost=0.43..0.62 rows=1 width=12) (actual time=0.033..0.033 rows=0 loops=540089)"

"   Index Cond: ((amka)::text = (js.""StudentAMKA"")::text)"

"   Filter: ((entry_date >= '2040-09-01'::date) AND (entry_date <= '2050-09-30'::date))"

"   Rows Removed by Filter: 1"

"   -> Index Only Scan using ""Student_pkey"" on ""Student"" s (cost=0.43..6.45 rows=1 width=12) (actual time=0.031..0.031 rows=1 loops=553)"

"   Index Cond: (amka = (js.""StudentAMKA"")::text)"

"   Heap Fetches: 0"

"   -> Index Scan using ""Person_pkey"" on ""Person"" p (cost=0.43..0.66 rows=1 width=16) (actual time=0.036..0.036 rows=1 loops=553)"

"   Index Cond: ((amka)::text = (s.amka)::text)"

"Planning Time: 1.136 ms"

"Execution Time: 7308.692 ms"

ΔΟΚΙΜΉ 2Η

"Sort (cost=84806.24..84807.35 rows=445 width=30) (actual time=6175.672..6179.848 rows=445 loops=1)"

" Sort Key: (CASE WHEN (c.population > '50000'::numeric) THEN COALESCE(counts.total_students, '0'::bigint) ELSE '0'::bigint END) DESC"

" Sort Method: quicksort Memory: 61kB"

" -> Hash Left Join (cost=84772.92..84786.66 rows=445 width=30) (actual time=6175.079..6179.689 rows=445 loops=1)"

"   Hash Cond: (c.id = counts.city_id)"

"   -> Seq Scan on ""Cities"" c (cost=0.00..11.45 rows=445 width=31) (actual time=0.016..0.196 rows=445 loops=1)"

"   -> Hash (cost=84772.90..84772.90 rows=2 width=12) (actual time=6175.053..6179.208 rows=28 loops=1)"

"   Buckets: 1024 Batches: 1 Memory Usage: 10kB"

"   -> Subquery Scan on counts (cost=84772.84..84772.90 rows=2 width=12) (actual time=6174.289..6179.190 rows=28 loops=1)"

"   -> GroupAggregate (cost=84772.84..84772.88 rows=2 width=12) (actual time=6174.275..6179.171 rows=28 loops=1)"

"   Group Key: p.city_id"

"   -> Sort (cost=84772.84..84772.85 rows=2 width=16) (actual time=6172.318..6176.511 rows=553 loops=1)"

"   Sort Key: p.city_id"

"   Sort Method: quicksort Memory: 50kB"

"   -> Nested Loop (cost=84758.87..84772.83 rows=2 width=16) (actual time=6006.939..6176.135 rows=553 loops=1)"

"   -> Nested Loop (cost=84758.44..84771.51 rows=2 width=24) (actual time=6006.880..6156.291 rows=553 loops=1)"

"   -> Finalize GroupAggregate (cost=84758.01..84758.60 rows=2 width=12) (actual time=6006.778..6136.497 rows=553 loops=1)"

"   Group Key: js.""StudentAMKA"""

"   Filter: (count(js.""ProgramID"") >= 2)"

"   Rows Removed by Filter: 58972"

"   -> Gather Merge (cost=84758.01..84758.50 rows=4 width=20) (actual time=6006.715..6108.249 rows=59528 loops=1)"

"   Workers Planned: 2"

"   Workers Launched: 2"

"   -> Partial GroupAggregate (cost=83757.98..83758.02 rows=2 width=20) (actual time=5932.271..5945.762 rows=19843 loops=3)"

"   Group Key: js.""StudentAMKA"""

" -> Sort (cost=83757.98..83757.99 rows=2 width=16) (actual time=5932.258..5933.766 rows=20027 loops=3)"

" Sort Key: js.""StudentAMKA"""

" Sort Method: quicksort Memory: 1729kB"

" Worker 0: Sort Method: quicksort Memory: 1714kB"

" Worker 1: Sort Method: quicksort Memory: 1678kB"

" -> Nested Loop (cost=10.37..83757.97 rows=2 width=16) (actual time=3.504..5903.474 rows=20027 loops=3)"

" -> Hash Join (cost=9.94..83715.88 rows=68 width=16) (actual time=2.374..551.503 rows=180030 loops=3)"

" Hash Cond: (js.""ProgramID"" = pgr.""ProgramID"")"

" -> Parallel Index Only Scan using ""Joins_pkey"" on ""Joins"" js (cost=0.43..81021.51 rows=1022792 width=16) (actual time=1.828..380.202 rows=821887 loops=3)"

" Heap Fetches: 500334"

" -> Hash (cost=9.10..9.10 rows=33 width=4) (actual time=0.456..0.457 rows=11 loops=3)"

" Buckets: 1024 Batches: 1 Memory Usage: 9kB"

" -> Index Scan using idx_program_duration on ""Program"" pgr (cost=0.42..9.10 rows=33 width=4) (actual time=0.437..0.445 rows=11 loops=3)"

" Index Cond: (""Duration"" = 5)"

" -> Index Scan using ""Student_pkey"" on ""Student"" (cost=0.43..0.62 rows=1 width=12) (actual time=0.029..0.029 rows=0 loops=540089)"

" Index Cond: ((amka)::text = (js.""StudentAMKA"")::text)"

" Filter: ((entry_date >= '2040-09-01'::date) AND (entry_date <= '2050-09-30'::date))"

" Rows Removed by Filter: 1"

" -> Index Only Scan using ""Student_pkey"" on ""Student"" s (cost=0.43..6.45 rows=1 width=12) (actual time=0.034..0.034 rows=1 loops=553)"

" Index Cond: (amka = (js.""StudentAMKA"")::text)"

" Heap Fetches: 0"

" -> Index Scan using ""Person_pkey"" on ""Person"" p (cost=0.43..0.66 rows=1 width=16) (actual time=0.035..0.035 rows=1 loops=553)"

" Index Cond: ((amka)::text = (s.amka)::text)"

"Planning Time: 1.287 ms"

"Execution Time: 6180.249 ms"

ΔΟΚΙΜΉ 3Η

"Sort (cost=84806.24..84807.35 rows=445 width=30) (actual time=6156.474..6161.180 rows=445 loops=1)"

" Sort Key: (CASE WHEN (c.population > '50000'::numeric) THEN COALESCE(counts.total_students, '0'::bigint) ELSE '0'::bigint END) DESC"

" Sort Method: quicksort Memory: 61kB"

" -> Hash Left Join (cost=84772.92..84786.66 rows=445 width=30) (actual time=6156.104..6161.064 rows=445 loops=1)"

"   Hash Cond: (c.id = counts.city_id)"

"   -> Seq Scan on ""Cities"" c (cost=0.00..11.45 rows=445 width=31) (actual time=0.012..0.121 rows=445 loops=1)"

"   -> Hash (cost=84772.90..84772.90 rows=2 width=12) (actual time=6156.082..6160.761 rows=28 loops=1)"

"   Buckets: 1024 Batches: 1 Memory Usage: 10kB"

"   -> Subquery Scan on counts (cost=84772.84..84772.90 rows=2 width=12) (actual time=6155.553..6160.746 rows=28 loops=1)"

"   -> GroupAggregate (cost=84772.84..84772.88 rows=2 width=12) (actual time=6155.552..6160.739 rows=28 loops=1)"

"   Group Key: p.city_id"

"   -> Sort (cost=84772.84..84772.85 rows=2 width=16) (actual time=6153.490..6158.228 rows=553 loops=1)"

"   Sort Key: p.city_id"

"   Sort Method: quicksort Memory: 50kB"

"   -> Nested Loop (cost=84758.87..84772.83 rows=2 width=16) (actual time=5982.956..6157.814 rows=553 loops=1)"

"   -> Nested Loop (cost=84758.44..84771.51 rows=2 width=24) (actual time=5982.871..6133.527 rows=553 loops=1)"

"   -> Finalize GroupAggregate (cost=84758.01..84758.60 rows=2 width=12) (actual time=5982.764..6115.207 rows=553 loops=1)"

"   Group Key: js.""StudentAMKA"""

"   Filter: (count(js.""ProgramID"") >= 2)"

"   Rows Removed by Filter: 58972"

"   -> Gather Merge (cost=84758.01..84758.50 rows=4 width=20) (actual time=5982.707..6086.656 rows=59528 loops=1)"

"   Workers Planned: 2"

"   Workers Launched: 2"

"   -> Partial GroupAggregate (cost=83757.98..83758.02 rows=2 width=20) (actual time=5912.075..5926.054 rows=19843 loops=3)"

"   Group Key: js.""StudentAMKA"""

" -> Sort (cost=83757.98..83757.99 rows=2 width=16) (actual time=5912.063..5913.448 rows=20027 loops=3)"

" Sort Key: js.""StudentAMKA"""

" Sort Method: quicksort Memory: 1741kB"

" Worker 0: Sort Method: quicksort Memory: 1699kB"

" Worker 1: Sort Method: quicksort Memory: 1682kB"

" -> Nested Loop (cost=10.37..83757.97 rows=2 width=16) (actual time=6.949..5884.323 rows=20027 loops=3)"

" -> Hash Join (cost=9.94..83715.88 rows=68 width=16) (actual time=2.081..559.553 rows=180030 loops=3)"

" Hash Cond: (js.""ProgramID"" = pgr.""ProgramID"")"

" -> Parallel Index Only Scan using ""Joins_pkey"" on ""Joins"" js (cost=0.43..81021.51 rows=1022792 width=16) (actual time=0.965..381.016 rows=821887 loops=3)"

" Heap Fetches: 500334"

" -> Hash (cost=9.10..9.10 rows=33 width=4) (actual time=0.666..0.667 rows=11 loops=3)"

" Buckets: 1024 Batches: 1 Memory Usage: 9kB"

" -> Index Scan using idx_program_duration on ""Program"" pgr (cost=0.42..9.10 rows=33 width=4) (actual time=0.643..0.651 rows=11 loops=3)"

" Index Cond: (""Duration"" = 5)"

" -> Index Scan using ""Student_pkey"" on ""Student"" (cost=0.43..0.62 rows=1 width=12) (actual time=0.029..0.029 rows=0 loops=540089)"

" Index Cond: ((amka)::text = (js.""StudentAMKA"")::text)"

" Filter: ((entry_date >= '2040-09-01'::date) AND (entry_date <= '2050-09-30'::date))"

" Rows Removed by Filter: 1"

" -> Index Only Scan using ""Student_pkey"" on ""Student"" s (cost=0.43..6.45 rows=1 width=12) (actual time=0.031..0.031 rows=1 loops=553)"

" Index Cond: (amka = (js.""StudentAMKA"")::text)"

" Heap Fetches: 0"

" -> Index Scan using ""Person_pkey"" on ""Person"" p (cost=0.43..0.66 rows=1 width=16) (actual time=0.042..0.042 rows=1 loops=553)"

" Index Cond: ((amka)::text = (s.amka)::text)"

"Planning Time: 2.697 ms"

"Execution Time: 6161.587 ms"

Query έπειτα από εισαγωγή δεδομένων και χρήση ευρετηρίων HASH:

Joins: StudentAMKA | Program: Duration

ΔΟΚΙΜΉ 1Η

"Sort (cost=84897.20..84898.31 rows=445 width=30) (actual time=6615.306..6615.439 rows=445 loops=1)"

" Sort Key: (CASE WHEN (c.population > '50000'::numeric) THEN COALESCE(counts.total_students, '0'::bigint) ELSE '0'::bigint END) DESC"

" Sort Method: quicksort Memory: 61kB"

" -> Hash Left Join (cost=84863.88..84877.62 rows=445 width=30) (actual time=6614.923..6615.312 rows=445 loops=1)"

"   Hash Cond: (c.id = counts.city_id)"

"   -> Seq Scan on ""Cities"" c (cost=0.00..11.45 rows=445 width=31) (actual time=0.012..0.147 rows=445 loops=1)"

"   -> Hash (cost=84863.86..84863.86 rows=2 width=12) (actual time=6614.898..6614.993 rows=28 loops=1)"

"   Buckets: 1024 Batches: 1 Memory Usage: 10kB"

"   -> Subquery Scan on counts (cost=84863.80..84863.86 rows=2 width=12) (actual time=6614.484..6614.972 rows=28 loops=1)"

"   -> GroupAggregate (cost=84863.80..84863.84 rows=2 width=12) (actual time=6614.483..6614.965 rows=28 loops=1)"

"   Group Key: p.city_id"

"   -> Sort (cost=84863.80..84863.81 rows=2 width=16) (actual time=6612.564..6612.696 rows=553 loops=1)"

"   Sort Key: p.city_id"

"   Sort Method: quicksort Memory: 50kB"

"   -> Nested Loop (cost=84849.83..84863.79 rows=2 width=16) (actual time=6449.013..6612.321 rows=553 loops=1)"

"   -> Nested Loop (cost=84849.40..84862.47 rows=2 width=24) (actual time=6448.549..6588.697 rows=553 loops=1)"

"   -> Finalize GroupAggregate (cost=84848.97..84849.56 rows=2 width=12) (actual time=6448.409..6566.305 rows=553 loops=1)"

"   Group Key: js.""StudentAMKA"""

"   Filter: (count(js.""ProgramID"") >= 2)"

"   Rows Removed by Filter: 58972"

"   -> Gather Merge (cost=84848.97..84849.46 rows=4 width=20) (actual time=6448.351..6538.324 rows=59526 loops=1)"

"   Workers Planned: 2"

"   Workers Launched: 2"

"   -> Partial GroupAggregate (cost=83848.94..83848.98 rows=2 width=20) (actual time=6343.061..6356.536 rows=19842 loops=3)"

"   Group Key: js.""StudentAMKA"""

"   -> Sort (cost=83848.94..83848.95 rows=2 width=16) (actual time=6343.042..6344.488 rows=20027 loops=3)"

"   Sort Key: js.""StudentAMKA"""

" Sort Method: quicksort Memory: 1722kB"

" Worker 0: Sort Method: quicksort Memory: 1681kB"

" Worker 1: Sort Method: quicksort Memory: 1719kB"

" -> Nested Loop (cost=130.27..83848.93 rows=2 width=16) (actual time=1.896..6300.295 rows=20027 loops=3)"

" -> Hash Join (cost=130.27..83836.21 rows=68 width=16) (actual time=1.272..608.801 rows=180030 loops=3)"

" Hash Cond: (js.""ProgramID"" = pgr.""ProgramID"")"

" -> Parallel Index Only Scan using ""Joins_pkey"" on ""Joins"" js (cost=0.43..81021.51 rows=1022792 width=16) (actual time=0.938..399.316 rows=821887 loops=3)"

" Heap Fetches: 500334"

" -> Hash (cost=129.43..129.43 rows=33 width=4) (actual time=0.251..0.253 rows=11 loops=3)"

" Buckets: 1024 Batches: 1 Memory Usage: 9kB"

" -> Bitmap Heap Scan on ""Program"" pgr (cost=4.26..129.43 rows=33 width=4) (actual time=0.231..0.240 rows=11 loops=3)"

" Recheck Cond: (""Duration"" = 5)"

" Heap Blocks: exact=2"

" -> Bitmap Index Scan on idx_program_duration (cost=0.00..4.25 rows=33 width=0) (actual time=0.215..0.215 rows=11 loops=3)"

" Index Cond: (""Duration"" = 5)"

" -> Index Scan using idx_student_amka on ""Student"" (cost=0.00..0.19 rows=1 width=12) (actual time=0.031..0.031 rows=0 loops=540089)"

" Index Cond: ((amka)::text = (js.""StudentAMKA"")::text)"

" Rows Removed by Index Recheck: 0"

" Filter: ((entry_date >= '2040-09-01'::date) AND (entry_date <= '2050-09-30'::date))"

" Rows Removed by Filter: 1"

" -> Index Only Scan using ""Student_pkey"" on ""Student"" s (cost=0.43..6.45 rows=1 width=12) (actual time=0.039..0.039 rows=1 loops=553)"

" Index Cond: (amka = (js.""StudentAMKA"")::text)"

" Heap Fetches: 0"

" -> Index Scan using ""Person_pkey"" on ""Person"" p (cost=0.43..0.66 rows=1 width=16) (actual time=0.041..0.041 rows=1 loops=553)"

" Index Cond: ((amka)::text = (s.amka)::text)"

"Planning Time: 1.372 ms" "Execution Time: 6615.894 ms"

ΔΟΚΙΜΉ 2Η

"Sort (cost=84897.20..84898.31 rows=445 width=30) (actual time=6801.650..6803.908 rows=445 loops=1)"

" Sort Key: (CASE WHEN (c.population > '50000'::numeric) THEN COALESCE(counts.total_students, '0'::bigint) ELSE '0'::bigint END) DESC"

" Sort Method: quicksort Memory: 61kB"

" -> Hash Left Join (cost=84863.88..84877.62 rows=445 width=30) (actual time=6801.283..6803.799 rows=445 loops=1)"

"   Hash Cond: (c.id = counts.city_id)"

"   -> Seq Scan on ""Cities"" c (cost=0.00..11.45 rows=445 width=31) (actual time=0.009..0.128 rows=445 loops=1)"

"   -> Hash (cost=84863.86..84863.86 rows=2 width=12) (actual time=6801.263..6803.500 rows=28 loops=1)"

"   Buckets: 1024 Batches: 1 Memory Usage: 10kB"

"   -> Subquery Scan on counts (cost=84863.80..84863.86 rows=2 width=12) (actual time=6798.836..6803.459 rows=28 loops=1)"

"   -> GroupAggregate (cost=84863.80..84863.84 rows=2 width=12) (actual time=6798.833..6803.448 rows=28 loops=1)"

"   Group Key: p.city_id"

"   -> Sort (cost=84863.80..84863.81 rows=2 width=16) (actual time=6796.970..6799.268 rows=553 loops=1)"

"   Sort Key: p.city_id"

"   Sort Method: quicksort Memory: 50kB"

"   -> Nested Loop (cost=84849.83..84863.79 rows=2 width=16) (actual time=6641.491..6798.825 rows=553 loops=1)"

"   -> Nested Loop (cost=84849.40..84862.47 rows=2 width=24) (actual time=6641.430..6777.626 rows=553 loops=1)"

"   -> Finalize GroupAggregate (cost=84848.97..84849.56 rows=2 width=12) (actual time=6641.303..6759.138 rows=553 loops=1)"

"   Group Key: js.""StudentAMKA"""

"   Filter: (count(js.""ProgramID"") >= 2)"

"   Rows Removed by Filter: 58972"

"   -> Gather Merge (cost=84848.97..84849.46 rows=4 width=20) (actual time=6641.240..6732.388 rows=59527 loops=1)"

"   Workers Planned: 2"

"   Workers Launched: 2"

"   -> Partial GroupAggregate (cost=83848.94..83848.98 rows=2 width=20) (actual time=6583.861..6596.618 rows=19842 loops=3)"

"   Group Key: js.""StudentAMKA"""

"   -> Sort (cost=83848.94..83848.95 rows=2 width=16) (actual time=6583.848..6585.119 rows=20027 loops=3)"

"   Sort Key: js.""StudentAMKA"""

"   Sort Method: quicksort Memory: 1726kB"

"   Worker 0: Sort Method: quicksort Memory: 1698kB"

" Worker 1: Sort Method: quicksort Memory: 1698kB"

" -> Nested Loop (cost=130.27..83848.93 rows=2 width=16) (actual time=6.972..6549.215 rows=20027 loops=3)"

" -> Hash Join (cost=130.27..83836.21 rows=68 width=16) (actual time=4.572..1069.653 rows=180030 loops=3)"

" Hash Cond: (js.""ProgramID"" = pgr.""ProgramID"")"

" -> Parallel Index Only Scan using ""Joins_pkey"" on ""Joins"" js (cost=0.43..81021.51 rows=1022792 width=16) (actual time=3.948..886.257 rows=821887 loops=3)"

" Heap Fetches: 500334"

" -> Hash (cost=129.43..129.43 rows=33 width=4) (actual time=0.193..0.195 rows=11 loops=3)"

" Buckets: 1024 Batches: 1 Memory Usage: 9kB"

" -> Bitmap Heap Scan on ""Program"" pgr (cost=4.26..129.43 rows=33 width=4) (actual time=0.174..0.182 rows=11 loops=3)"

" Recheck Cond: (""Duration"" = 5)"

" Heap Blocks: exact=2"

" -> Bitmap Index Scan on idx_program_duration (cost=0.00..4.25 rows=33 width=0) (actual time=0.164..0.164 rows=11 loops=3)"

" Index Cond: (""Duration"" = 5)"

" -> Index Scan using idx_student_amka on ""Student"" (cost=0.00..0.19 rows=1 width=12) (actual time=0.030..0.030 rows=0 loops=540089)"

" Index Cond: ((amka)::text = (js.""StudentAMKA"")::text)"

" Rows Removed by Index Recheck: 0"

" Filter: ((entry_date >= '2040-09-01'::date) AND (entry_date <= '2050-09-30'::date))"

" Rows Removed by Filter: 1"

" -> Index Only Scan using ""Student_pkey"" on ""Student"" s (cost=0.43..6.45 rows=1 width=12) (actual time=0.032..0.032 rows=1 loops=553)"

" Index Cond: (amka = (js.""StudentAMKA"")::text)"

" Heap Fetches: 0"

" -> Index Scan using ""Person_pkey"" on ""Person"" p (cost=0.43..0.66 rows=1 width=16) (actual time=0.037..0.037 rows=1 loops=553)"

" Index Cond: ((amka)::text = (s.amka)::text)"

"Planning Time: 2.901 ms"

"Execution Time: 6804.382 ms"

ΔΟΚΙΜΉ 3Η

"Sort (cost=84897.20..84898.31 rows=445 width=30) (actual time=6107.693..6114.115 rows=445 loops=1)"

" Sort Key: (CASE WHEN (c.population > '50000'::numeric) THEN COALESCE(counts.total_students, '0'::bigint) ELSE '0'::bigint END) DESC"

" Sort Method: quicksort Memory: 61kB"

" -> Hash Left Join (cost=84863.88..84877.62 rows=445 width=30) (actual time=6107.354..6114.008 rows=445 loops=1)"

"   Hash Cond: (c.id = counts.city_id)"

"   -> Seq Scan on ""Cities"" c (cost=0.00..11.45 rows=445 width=31) (actual time=0.008..0.109 rows=445 loops=1)"

"   -> Hash (cost=84863.86..84863.86 rows=2 width=12) (actual time=6107.337..6113.733 rows=28 loops=1)"

"   Buckets: 1024 Batches: 1 Memory Usage: 10kB"

"   -> Subquery Scan on counts (cost=84863.80..84863.86 rows=2 width=12) (actual time=6106.940..6113.712 rows=28 loops=1)"

"   -> GroupAggregate (cost=84863.80..84863.84 rows=2 width=12) (actual time=6106.939..6113.707 rows=28 loops=1)"

"   Group Key: p.city_id"

"   -> Sort (cost=84863.80..84863.81 rows=2 width=16) (actual time=6105.141..6111.566 rows=553 loops=1)"

"   Sort Key: p.city_id"

"   Sort Method: quicksort Memory: 50kB"

"   -> Nested Loop (cost=84849.83..84863.79 rows=2 width=16) (actual time=5951.074..6111.052 rows=553 loops=1)"

"   -> Nested Loop (cost=84849.40..84862.47 rows=2 width=24) (actual time=5951.008..6090.463 rows=553 loops=1)"

"   -> Finalize GroupAggregate (cost=84848.97..84849.56 rows=2 width=12) (actual time=5950.901..6072.559 rows=553 loops=1)"

"   Group Key: js.""StudentAMKA"""

"   Filter: (count(js.""ProgramID"") >= 2)"

"   Rows Removed by Filter: 58972"

"   -> Gather Merge (cost=84848.97..84849.46 rows=4 width=20) (actual time=5950.838..6044.185 rows=59528 loops=1)"

"   Workers Planned: 2"

"   Workers Launched: 2"

"   -> Partial GroupAggregate (cost=83848.94..83848.98 rows=2 width=20) (actual time=5852.024..5865.347 rows=19843 loops=3)"

"   Group Key: js.""StudentAMKA"""

"   -> Sort (cost=83848.94..83848.95 rows=2 width=16) (actual time=5852.012..5853.326 rows=20027 loops=3)"

"   Sort Key: js.""StudentAMKA"""

"   Sort Method: quicksort Memory: 1732kB"

"   Worker 0: Sort Method: quicksort Memory: 1712kB"

" Worker 1: Sort Method: quicksort Memory: 1678kB"

" -> Nested Loop (cost=130.27..83848.93 rows=2 width=16) (actual time=2.729..5821.673 rows=20027 loops=3)"

" -> Hash Join (cost=130.27..83836.21 rows=68 width=16) (actual time=1.519..602.473 rows=180030 loops=3)"

" Hash Cond: (js.""ProgramID"" = pgr.""ProgramID"")"

" -> Parallel Index Only Scan using ""Joins_pkey"" on ""Joins"" js (cost=0.43..81021.51 rows=1022792 width=16) (actual time=1.104..427.944 rows=821887 loops=3)"

" Heap Fetches: 500334"

" -> Hash (cost=129.43..129.43 rows=33 width=4) (actual time=0.319..0.321 rows=11 loops=3)"

" Buckets: 1024 Batches: 1 Memory Usage: 9kB"

" -> Bitmap Heap Scan on ""Program"" pgr (cost=4.26..129.43 rows=33 width=4) (actual time=0.292..0.304 rows=11 loops=3)"

" Recheck Cond: (""Duration"" = 5)"

" Heap Blocks: exact=2"

" -> Bitmap Index Scan on idx_program_duration (cost=0.00..4.25 rows=33 width=0) (actual time=0.278..0.278 rows=11 loops=3)"

" Index Cond: (""Duration"" = 5)"

" -> Index Scan using idx_student_amka on ""Student"" (cost=0.00..0.19 rows=1 width=12) (actual time=0.028..0.028 rows=0 loops=540089)"

" Index Cond: ((amka)::text = (js.""StudentAMKA"")::text)"

" Rows Removed by Index Recheck: 0"

" Filter: ((entry_date >= '2040-09-01'::date) AND (entry_date <= '2050-09-30'::date))"

" Rows Removed by Filter: 1"

" -> Index Only Scan using ""Student_pkey"" on ""Student"" s (cost=0.43..6.45 rows=1 width=12) (actual time=0.031..0.031 rows=1 loops=553)"

" Index Cond: (amka = (js.""StudentAMKA"")::text)"

" Heap Fetches: 0"

" -> Index Scan using ""Person_pkey"" on ""Person"" p (cost=0.43..0.66 rows=1 width=16) (actual time=0.036..0.036 rows=1 loops=553)"

" Index Cond: ((amka)::text = (s.amka)::text)"

"Planning Time: 1.538 ms"

"Execution Time: 6114.540 ms"

Query έπειτα από εισαγωγή δεδομένων και χρήση βέλτιστων ευρετηρίων μαζί με τα αντίστοιχα clusters.

HASH: Program: Duration | B-Tree: Student: entry_date | clusters και των δύο

ΔΟΚΙΜΉ 1Η

"Sort (cost=84930.44..84931.56 rows=445 width=30) (actual time=6157.384..6157.796 rows=445 loops=1)"

" Sort Key: (CASE WHEN (c.population > '50000'::numeric) THEN COALESCE(counts.total_students, '0'::bigint) ELSE '0'::bigint END) DESC"

" Sort Method: quicksort Memory: 61kB"

" -> Hash Left Join (cost=84897.13..84910.87 rows=445 width=30) (actual time=6157.044..6157.694 rows=445 loops=1)"

"   Hash Cond: (c.id = counts.city_id)"

"   -> Seq Scan on ""Cities"" c (cost=0.00..11.45 rows=445 width=31) (actual time=0.007..0.108 rows=445 loops=1)"

"   -> Hash (cost=84897.10..84897.10 rows=2 width=12) (actual time=6157.027..6157.418 rows=28 loops=1)"

"   Buckets: 1024 Batches: 1 Memory Usage: 10kB"

"   -> Subquery Scan on counts (cost=84897.05..84897.10 rows=2 width=12) (actual time=6156.337..6157.396 rows=28 loops=1)"

"   -> GroupAggregate (cost=84897.05..84897.08 rows=2 width=12) (actual time=6156.337..6157.389 rows=28 loops=1)"

"   Group Key: p.city_id"

"   -> Sort (cost=84897.05..84897.05 rows=2 width=16) (actual time=6154.546..6154.969 rows=553 loops=1)"

"   Sort Key: p.city_id"

"   Sort Method: quicksort Memory: 50kB"

"   -> Nested Loop (cost=84879.20..84897.04 rows=2 width=16) (actual time=5993.061..6154.555 rows=553 loops=1)"

"   -> Nested Loop (cost=84878.77..84895.84 rows=2 width=28) (actual time=5993.021..6134.035 rows=553 loops=1)"

"   -> Finalize GroupAggregate (cost=84878.34..84878.93 rows=2 width=12) (actual time=5992.909..6109.866 rows=553 loops=1)"

"   Group Key: js.""StudentAMKA"""

"   Filter: (count(js.""ProgramID"") >= 2)"

"   Rows Removed by Filter: 58972"

"   -> Gather Merge (cost=84878.34..84878.83 rows=4 width=20) (actual time=5992.847..6081.812 rows=59528 loops=1)"

"   Workers Planned: 2"

"   Workers Launched: 2"

"   -> Partial GroupAggregate (cost=83878.31..83878.35 rows=2 width=20) (actual time=5927.168..5940.735 rows=19843 loops=3)"

"   Group Key: js.""StudentAMKA"""

"   -> Sort (cost=83878.31..83878.32 rows=2 width=16) (actual time=5927.155..5928.682 rows=20027 loops=3)"

"   Sort Key: js.""StudentAMKA"""

"   Sort Method: quicksort Memory: 1730kB"

"   Worker 0: Sort Method: quicksort Memory: 1687kB"

"   Worker 1: Sort Method: quicksort Memory: 1705kB"

"   -> Nested Loop (cost=130.70..83878.30 rows=2 width=16) (actual time=1.666..5898.115 rows=20027 loops=3)"

"   -> Hash Join (cost=130.27..83836.21 rows=68 width=16) (actual time=0.797..555.272 rows=180030 loops=3)"

"   Hash Cond: (js.""ProgramID"" = pgr.""ProgramID"")"

"   -> Parallel Index Only Scan using ""Joins_pkey"" on ""Joins"" js (cost=0.43..81021.51 rows=1022792 width=16) (actual time=0.547..391.653 rows=821887 loops=3)"

"   Heap Fetches: 500334"

"   -> Hash (cost=129.43..129.43 rows=33 width=4) (actual time=0.169..0.171 rows=11 loops=3)"

"   Buckets: 1024 Batches: 1 Memory Usage: 9kB"

"   -> Bitmap Heap Scan on ""Program"" pgr (cost=4.26..129.43 rows=33 width=4) (actual time=0.154..0.161 rows=11 loops=3)"

"   Recheck Cond: (""Duration"" = 5)"

"   Heap Blocks: exact=2"

"   -> Bitmap Index Scan on idx_program_duration (cost=0.00..4.25 rows=33 width=0) (actual time=0.143..0.143 rows=11 loops=3)"

"   Index Cond: (""Duration"" = 5)"

"   -> Index Scan using ""Student_pkey"" on ""Student"" (cost=0.43..0.62 rows=1 width=12) (actual time=0.029..0.029 rows=0 loops=540089)"

"   Index Cond: ((amka)::text = (js.""StudentAMKA"")::text)"

"   Filter: ((entry_date >= '2040-09-01'::date) AND (entry_date <= '2050-09-30'::date))"

"   Rows Removed by Filter: 1"

"   -> Index Scan using ""Person_pkey"" on ""Person"" p (cost=0.43..8.45 rows=1 width=16) (actual time=0.042..0.042 rows=1 loops=553)"

"   Index Cond: ((amka)::text = (js.""StudentAMKA"")::text)"

"   -> Index Only Scan using ""Student_pkey"" on ""Student"" s (cost=0.43..0.60 rows=1 width=12) (actual time=0.036..0.036 rows=1 loops=553)"

"   Index Cond: (amka = (p.amka)::text)"

"   Heap Fetches: 553"

"Planning Time: 0.992 ms"

"Execution Time: 6158.172 ms"

ΔΟΚΙΜΉ 2Η

"Sort (cost=84930.44..84931.56 rows=445 width=30) (actual time=6319.124..6319.519 rows=445 loops=1)"

" Sort Key: (CASE WHEN (c.population > '50000'::numeric) THEN COALESCE(counts.total_students, '0'::bigint) ELSE '0'::bigint END) DESC"

" Sort Method: quicksort Memory: 61kB"

" -> Hash Left Join (cost=84897.13..84910.87 rows=445 width=30) (actual time=6318.773..6319.433 rows=445 loops=1)"

"   Hash Cond: (c.id = counts.city_id)"

"   -> Seq Scan on ""Cities"" c (cost=0.00..11.45 rows=445 width=31) (actual time=0.010..0.121 rows=445 loops=1)"

"   -> Hash (cost=84897.10..84897.10 rows=2 width=12) (actual time=6318.752..6319.146 rows=28 loops=1)"

"   Buckets: 1024 Batches: 1 Memory Usage: 10kB"

"   -> Subquery Scan on counts (cost=84897.05..84897.10 rows=2 width=12) (actual time=6317.783..6319.105 rows=28 loops=1)"

"   -> GroupAggregate (cost=84897.05..84897.08 rows=2 width=12) (actual time=6317.781..6319.095 rows=28 loops=1)"

"   Group Key: p.city_id"

"   -> Sort (cost=84897.05..84897.05 rows=2 width=16) (actual time=6313.316..6313.987 rows=553 loops=1)"

"   Sort Key: p.city_id"

"   Sort Method: quicksort Memory: 50kB"

"   -> Nested Loop (cost=84879.20..84897.04 rows=2 width=16) (actual time=6154.140..6313.393 rows=553 loops=1)"

"   -> Nested Loop (cost=84878.77..84895.84 rows=2 width=28) (actual time=6154.090..6294.592 rows=553 loops=1)"

"   -> Finalize GroupAggregate (cost=84878.34..84878.93 rows=2 width=12) (actual time=6153.918..6268.586 rows=553 loops=1)"

"   Group Key: js.""StudentAMKA"""

"   Filter: (count(js.""ProgramID"") >= 2)"

"   Rows Removed by Filter: 58972"

"   -> Gather Merge (cost=84878.34..84878.83 rows=4 width=20) (actual time=6153.857..6241.046 rows=59528 loops=1)"

"   Workers Planned: 2"

"   Workers Launched: 2"

"   -> Partial GroupAggregate (cost=83878.31..83878.35 rows=2 width=20) (actual time=6086.414..6099.990 rows=19843 loops=3)"

"   Group Key: js.""StudentAMKA"""

"   -> Sort (cost=83878.31..83878.32 rows=2 width=16) (actual time=6086.400..6087.854 rows=20027 loops=3)"

"   Sort Key: js.""StudentAMKA"""

"   Sort Method: quicksort Memory: 1713kB"

"   Worker 0: Sort Method: quicksort Memory: 1708kB"

"   Worker 1: Sort Method: quicksort Memory: 1701kB"

"   -> Nested Loop (cost=130.70..83878.30 rows=2 width=16) (actual time=3.924..6056.625 rows=20027 loops=3)"

"   -> Hash Join (cost=130.27..83836.21 rows=68 width=16) (actual time=2.568..556.364 rows=180030 loops=3)"

"   Hash Cond: (js.""ProgramID"" = pgr.""ProgramID"")"

"   -> Parallel Index Only Scan using ""Joins_pkey"" on ""Joins"" js (cost=0.43..81021.51 rows=1022792 width=16) (actual time=2.165..384.237 rows=821887 loops=3)"

"   Heap Fetches: 500334"

"   -> Hash (cost=129.43..129.43 rows=33 width=4) (actual time=0.265..0.267 rows=11 loops=3)"

"   Buckets: 1024 Batches: 1 Memory Usage: 9kB"

"   -> Bitmap Heap Scan on ""Program"" pgr (cost=4.26..129.43 rows=33 width=4) (actual time=0.245..0.251 rows=11 loops=3)"

"   Recheck Cond: (""Duration"" = 5)"

"   Heap Blocks: exact=2"

"   -> Bitmap Index Scan on idx_program_duration (cost=0.00..4.25 rows=33 width=0) (actual time=0.228..0.228 rows=11 loops=3)"

"   Index Cond: (""Duration"" = 5)"

"   -> Index Scan using ""Student_pkey"" on ""Student"" (cost=0.43..0.62 rows=1 width=12) (actual time=0.030..0.030 rows=0 loops=540089)"

"   Index Cond: ((amka)::text = (js.""StudentAMKA"")::text)"

"   Filter: ((entry_date >= '2040-09-01'::date) AND (entry_date <= '2050-09-30'::date))"

"   Rows Removed by Filter: 1"

"   -> Index Scan using ""Person_pkey"" on ""Person"" p (cost=0.43..8.45 rows=1 width=16) (actual time=0.045..0.045 rows=1 loops=553)"

"   Index Cond: ((amka)::text = (js.""StudentAMKA"")::text)"

"   -> Index Only Scan using ""Student_pkey"" on ""Student"" s (cost=0.43..0.60 rows=1 width=12) (actual time=0.033..0.033 rows=1 loops=553)"

"   Index Cond: (amka = (p.amka)::text)"

"   Heap Fetches: 553"

"Planning Time: 1.221 ms"

"Execution Time: 6320.137 ms"

ΔΟΚΙΜΉ 3Η

"Sort (cost=84930.44..84931.56 rows=445 width=30) (actual time=6163.767..6171.630 rows=445 loops=1)"

" Sort Key: (CASE WHEN (c.population > '50000'::numeric) THEN COALESCE(counts.total_students, '0'::bigint) ELSE '0'::bigint END) DESC"

" Sort Method: quicksort Memory: 61kB"

" -> Hash Left Join (cost=84897.13..84910.87 rows=445 width=30) (actual time=6163.448..6171.527 rows=445 loops=1)"

"   Hash Cond: (c.id = counts.city_id)"

"   -> Seq Scan on ""Cities"" c (cost=0.00..11.45 rows=445 width=31) (actual time=0.009..0.094 rows=445 loops=1)"

"   -> Hash (cost=84897.10..84897.10 rows=2 width=12) (actual time=6163.429..6171.267 rows=28 loops=1)"

"   Buckets: 1024 Batches: 1 Memory Usage: 10kB"

"   -> Subquery Scan on counts (cost=84897.05..84897.10 rows=2 width=12) (actual time=6163.036..6171.245 rows=28 loops=1)"

"   -> GroupAggregate (cost=84897.05..84897.08 rows=2 width=12) (actual time=6163.035..6171.239 rows=28 loops=1)"

"   Group Key: p.city_id"

"   -> Sort (cost=84897.05..84897.05 rows=2 width=16) (actual time=6161.249..6169.113 rows=553 loops=1)"

"   Sort Key: p.city_id"

"   Sort Method: quicksort Memory: 50kB"

"   -> Nested Loop (cost=84879.20..84897.04 rows=2 width=16) (actual time=5991.591..6168.596 rows=553 loops=1)"

"   -> Nested Loop (cost=84878.77..84895.84 rows=2 width=28) (actual time=5991.501..6150.833 rows=553 loops=1)"

"   -> Finalize GroupAggregate (cost=84878.34..84878.93 rows=2 width=12) (actual time=5991.349..6126.321 rows=553 loops=1)"

"   Group Key: js.""StudentAMKA"""

"   Filter: (count(js.""ProgramID"") >= 2)"

"   Rows Removed by Filter: 58972"

"   -> Gather Merge (cost=84878.34..84878.83 rows=4 width=20) (actual time=5991.289..6097.130 rows=59528 loops=1)"

"   Workers Planned: 2"

"   Workers Launched: 2"

"   -> Partial GroupAggregate (cost=83878.31..83878.35 rows=2 width=20) (actual time=5917.170..5931.402 rows=19843 loops=3)"

"   Group Key: js.""StudentAMKA"""

"   -> Sort (cost=83878.31..83878.32 rows=2 width=16) (actual time=5917.156..5918.817 rows=20027 loops=3)"

"   Sort Key: js.""StudentAMKA"""

"   Sort Method: quicksort Memory: 1731kB"

"   Worker 0: Sort Method: quicksort Memory: 1719kB"

"   Worker 1: Sort Method: quicksort Memory: 1671kB"

"   -> Nested Loop (cost=130.70..83878.30 rows=2 width=16) (actual time=4.180..5887.433 rows=20027 loops=3)"

"   -> Hash Join (cost=130.27..83836.21 rows=68 width=16) (actual time=2.897..559.396 rows=180030 loops=3)"

"   Hash Cond: (js.""ProgramID"" = pgr.""ProgramID"")"

"   -> Parallel Index Only Scan using ""Joins_pkey"" on ""Joins"" js (cost=0.43..81021.51 rows=1022792 width=16) (actual time=1.909..389.702 rows=821887 loops=3)"

"   Heap Fetches: 500334"

"   -> Hash (cost=129.43..129.43 rows=33 width=4) (actual time=0.529..0.530 rows=11 loops=3)"

"   Buckets: 1024 Batches: 1 Memory Usage: 9kB"

"   -> Bitmap Heap Scan on ""Program"" pgr (cost=4.26..129.43 rows=33 width=4) (actual time=0.502..0.511 rows=11 loops=3)"

"   Recheck Cond: (""Duration"" = 5)"

"   Heap Blocks: exact=2"

"   -> Bitmap Index Scan on idx_program_duration (cost=0.00..4.25 rows=33 width=0) (actual time=0.411..0.411 rows=11 loops=3)"

"   Index Cond: (""Duration"" = 5)"

"   -> Index Scan using ""Student_pkey"" on ""Student"" (cost=0.43..0.62 rows=1 width=12) (actual time=0.029..0.029 rows=0 loops=540089)"

"   Index Cond: ((amka)::text = (js.""StudentAMKA"")::text)"

"   Filter: ((entry_date >= '2040-09-01'::date) AND (entry_date <= '2050-09-30'::date))"

"   Rows Removed by Filter: 1"

"   -> Index Scan using ""Person_pkey"" on ""Person"" p (cost=0.43..8.45 rows=1 width=16) (actual time=0.043..0.043 rows=1 loops=553)"

"   Index Cond: ((amka)::text = (js.""StudentAMKA"")::text)"

"   -> Index Only Scan using ""Student_pkey"" on ""Student"" s (cost=0.43..0.60 rows=1 width=12) (actual time=0.031..0.031 rows=1 loops=553)"

"   Index Cond: (amka = (p.amka)::text)"

"   Heap Fetches: 553"

"Planning Time: 1.209 ms"

"Execution Time: 6172.056 ms"

Query με αλλαγμένη σειρά join , έπειτα από εισαγωγή δεδομένων και χρήση βέλτιστων ευρετηρίων μαζί με τα αντίστοιχα clusters.

HASH: Program: Duration | B-Tree: Student: entry_date | clusters και των δύο

ΔΟΚΙΜΉ 1^{Η}

"Sort (cost=84930.44..84931.56 rows=445 width=30) (actual time=15422.492..15422.669 rows=445 loops=1)"

" Sort Key: (CASE WHEN (c.population > '50000'::numeric) THEN COALESCE(counts.total_students, '0'::bigint) ELSE '0'::bigint END) DESC"

" Sort Method: quicksort Memory: 61kB"

" -> Hash Left Join (cost=84897.13..84910.87 rows=445 width=30) (actual time=15422.055..15422.473 rows=445 loops=1)"

"   Hash Cond: (c.id = counts.city_id)"

"   -> Seq Scan on ""Cities"" c (cost=0.00..11.45 rows=445 width=31) (actual time=0.020..0.132 rows=445 loops=1)"

"   -> Hash (cost=84897.10..84897.10 rows=2 width=12) (actual time=15421.957..15422.105 rows=28 loops=1)"

"   Buckets: 1024 Batches: 1 Memory Usage: 10kB"

"   -> Subquery Scan on counts (cost=84897.05..84897.10 rows=2 width=12) (actual time=15421.562..15422.084 rows=28 loops=1)"

"   -> GroupAggregate (cost=84897.05..84897.08 rows=2 width=12) (actual time=15421.561..15422.078 rows=28 loops=1)"

"   Group Key: p.city_id"

"   -> Sort (cost=84897.05..84897.05 rows=2 width=16) (actual time=15419.706..15419.884 rows=553 loops=1)"

"   Sort Key: p.city_id"

"   Sort Method: quicksort Memory: 50kB"

"   -> Nested Loop (cost=84879.20..84897.04 rows=2 width=16) (actual time=14673.646..15418.189 rows=553 loops=1)"

"   -> Nested Loop (cost=84878.77..84895.84 rows=2 width=28) (actual time=14672.894..15187.305 rows=553 loops=1)"

"   -> Finalize GroupAggregate (cost=84878.34..84878.93 rows=2 width=12) (actual time=14669.294..14783.716 rows=553 loops=1)"

"   Group Key: js.""StudentAMKA"""

"   Filter: (count(js.""ProgramID"") >= 2)"

"   Rows Removed by Filter: 58972"

"   -> Gather Merge (cost=84878.34..84878.83 rows=4 width=20) (actual time=14669.238..14754.271 rows=59528 loops=1)"

"   Workers Planned: 2"

"   Workers Launched: 2"

"   -> Partial GroupAggregate (cost=83878.31..83878.35 rows=2 width=20) (actual time=14562.759..14575.626 rows=19843 loops=3)"

"   Group Key: js.""StudentAMKA"""

"   -> Sort (cost=83878.31..83878.32 rows=2 width=16) (actual time=14562.744..14564.288 rows=20027 loops=3)"

" Sort Key: js.""StudentAMKA"""

" Sort Method: quicksort Memory: 1710kB"

" Worker 0: Sort Method: quicksort Memory: 1692kB"

" Worker 1: Sort Method: quicksort Memory: 1720kB"

" -> Nested Loop (cost=130.70..83878.30 rows=2 width=16) (actual time=13.648..14515.848 rows=20027 loops=3)"

" -> Hash Join (cost=130.27..83836.21 rows=68 width=16) (actual time=4.413..2312.533 rows=180030 loops=3)"

" Hash Cond: (js.""ProgramID"" = pgr.""ProgramID"")"

" -> Parallel Index Only Scan using ""Joins_pkey"" on ""Joins"" js (cost=0.43..81021.51 rows=1022792 width=16) (actual time=2.802..2086.998 rows=821887 loops=3)"

" Heap Fetches: 500334"

" -> Hash (cost=129.43..129.43 rows=33 width=4) (actual time=1.291..1.293 rows=11 loops=3)"

" Buckets: 1024 Batches: 1 Memory Usage: 9kB"

" -> Bitmap Heap Scan on ""Program"" pgr (cost=4.26..129.43 rows=33 width=4) (actual time=1.239..1.273 rows=11 loops=3)"

" Recheck Cond: (""Duration"" = 5)"

" Heap Blocks: exact=2"

" -> Bitmap Index Scan on idx_program_duration (cost=0.00..4.25 rows=33 width=0) (actual time=1.013..1.013 rows=11 loops=3)"

" Index Cond: (""Duration"" = 5)"

" -> Index Scan using ""Student_pkey"" on ""Student"" (cost=0.43..0.62 rows=1 width=12) (actual time=0.067..0.067 rows=0 loops=540089)"

" Index Cond: ((amka)::text = (js.""StudentAMKA"")::text)"

" Filter: ((entry_date >= '2040-09-01'::date) AND (entry_date <= '2050-09-30'::date))"

" Rows Removed by Filter: 1"

" -> Index Scan using ""Person_pkey"" on ""Person"" p (cost=0.43..8.45 rows=1 width=16) (actual time=0.724..0.724 rows=1 loops=553)"

" Index Cond: ((amka)::text = (js.""StudentAMKA"")::text)"

" -> Index Only Scan using ""Student_pkey"" on ""Student"" s (cost=0.43..0.60 rows=1 width=12) (actual time=0.410..0.410 rows=1 loops=553)"

" Index Cond: (amka = (p.amka)::text)"

" Heap Fetches: 553"

"Planning Time: 9.728 ms"

"Execution Time: 15423.514 ms"

ΔΟΚΙΜΉ 2Η

"Sort (cost=84930.44..84931.56 rows=445 width=30) (actual time=12963.641..12963.744 rows=445 loops=1)"

" Sort Key: (CASE WHEN (c.population > '50000'::numeric) THEN COALESCE(counts.total_students, '0'::bigint) ELSE '0'::bigint END) DESC"

" Sort Method: quicksort Memory: 61kB"

" -> Hash Left Join (cost=84897.13..84910.87 rows=445 width=30) (actual time=12963.325..12963.637 rows=445 loops=1)"

"   Hash Cond: (c.id = counts.city_id)"

"   -> Seq Scan on ""Cities"" c (cost=0.00..11.45 rows=445 width=31) (actual time=0.010..0.089 rows=445 loops=1)"

"   -> Hash (cost=84897.10..84897.10 rows=2 width=12) (actual time=12963.306..12963.385 rows=28 loops=1)"

"   Buckets: 1024 Batches: 1 Memory Usage: 10kB"

"   -> Subquery Scan on counts (cost=84897.05..84897.10 rows=2 width=12) (actual time=12962.910..12963.362 rows=28 loops=1)"

"   -> GroupAggregate (cost=84897.05..84897.08 rows=2 width=12) (actual time=12962.909..12963.356 rows=28 loops=1)"

"   Group Key: p.city_id"

"   -> Sort (cost=84897.05..84897.05 rows=2 width=16) (actual time=12961.099..12961.208 rows=553 loops=1)"

"   Sort Key: p.city_id"

"   Sort Method: quicksort Memory: 50kB"

"   -> Nested Loop (cost=84879.20..84897.04 rows=2 width=16) (actual time=12258.922..12959.871 rows=553 loops=1)"

"   -> Nested Loop (cost=84878.77..84895.84 rows=2 width=28) (actual time=12258.104..12752.010 rows=553 loops=1)"

"   -> Finalize GroupAggregate (cost=84878.34..84878.93 rows=2 width=12) (actual time=12253.409..12366.387 rows=553 loops=1)"

"   Group Key: js.""StudentAMKA"""

"   Filter: (count(js.""ProgramID"") >= 2)"

"   Rows Removed by Filter: 58972"

"   -> Gather Merge (cost=84878.34..84878.83 rows=4 width=20) (actual time=12253.293..12337.319 rows=59528 loops=1)"

"   Workers Planned: 2"

"   Workers Launched: 2"

"   -> Partial GroupAggregate (cost=83878.31..83878.35 rows=2 width=20) (actual time=12167.554..12180.837 rows=19843 loops=3)"

"   Group Key: js.""StudentAMKA"""

"   -> Sort (cost=83878.31..83878.32 rows=2 width=16) (actual time=12167.540..12168.937 rows=20027 loops=3)"

"   Sort Key: js.""StudentAMKA"""

"   Sort Method: quicksort Memory: 1699kB"

"   Worker 0: Sort Method: quicksort Memory: 1718kB"

" Worker 1: Sort Method: quicksort Memory: 1706kB"

" -> Nested Loop (cost=130.70..83878.30 rows=2 width=16) (actual time=8.214..12105.579 rows=20027 loops=3)"

" -> Hash Join (cost=130.27..83836.21 rows=68 width=16) (actual time=3.493..2336.599 rows=180030 loops=3)"

" Hash Cond: (js.""ProgramID"" = pgr.""ProgramID"")"

" -> Parallel Index Only Scan using ""Joins_pkey"" on ""Joins"" js (cost=0.43..81021.51 rows=1022792 width=16) (actual time=3.029..2121.774 rows=821887 loops=3)"

" Heap Fetches: 500334"

" -> Hash (cost=129.43..129.43 rows=33 width=4) (actual time=0.345..0.347 rows=11 loops=3)"

" Buckets: 1024 Batches: 1 Memory Usage: 9kB"

" -> Bitmap Heap Scan on ""Program"" pgr (cost=4.26..129.43 rows=33 width=4) (actual time=0.325..0.332 rows=11 loops=3)"

" Recheck Cond: (""Duration"" = 5)"

" Heap Blocks: exact=2"

" -> Bitmap Index Scan on idx_program_duration (cost=0.00..4.25 rows=33 width=0) (actual time=0.308..0.308 rows=11 loops=3)"

" Index Cond: (""Duration"" = 5)"

" -> Index Scan using ""Student_pkey"" on ""Student"" (cost=0.43..0.62 rows=1 width=12) (actual time=0.053..0.053 rows=0 loops=540089)"

" Index Cond: ((amka)::text = (js.""StudentAMKA"")::text)"

" Filter: ((entry_date >= '2040-09-01'::date) AND (entry_date <= '2050-09-30'::date))"

" Rows Removed by Filter: 1"

" -> Index Scan using ""Person_pkey"" on ""Person"" p (cost=0.43..8.45 rows=1 width=16) (actual time=0.692..0.692 rows=1 loops=553)"

" Index Cond: ((amka)::text = (js.""StudentAMKA"")::text)"

" -> Index Only Scan using ""Student_pkey"" on ""Student"" s (cost=0.43..0.60 rows=1 width=12) (actual time=0.369..0.369 rows=1 loops=553)"

" Index Cond: (amka = (p.amka)::text)"

" Heap Fetches: 553"

"Planning Time: 4.099 ms"

"Execution Time: 12964.162 ms"

ΔΟΚΙΜΉΗ 3Η

"Sort (cost=84930.44..84931.56 rows=445 width=30) (actual time=10534.673..10534.798 rows=445 loops=1)"

" Sort Key: (CASE WHEN (c.population > '50000'::numeric) THEN COALESCE(counts.total_students, '0'::bigint) ELSE '0'::bigint END) DESC"

" Sort Method: quicksort Memory: 61kB"

" -> Hash Left Join (cost=84897.13..84910.87 rows=445 width=30) (actual time=10531.464..10533.681 rows=445 loops=1)"

"   Hash Cond: (c.id = counts.city_id)"

"   -> Seq Scan on ""Cities"" c (cost=0.00..11.45 rows=445 width=31) (actual time=0.010..1.917 rows=445 loops=1)"

"   -> Hash (cost=84897.10..84897.10 rows=2 width=12) (actual time=10530.727..10530.827 rows=28 loops=1)"

"   Buckets: 1024 Batches: 1 Memory Usage: 10kB"

"   -> Subquery Scan on counts (cost=84897.05..84897.10 rows=2 width=12) (actual time=10530.329..10530.808 rows=28 loops=1)"

"   -> GroupAggregate (cost=84897.05..84897.08 rows=2 width=12) (actual time=10530.326..10530.798 rows=28 loops=1)"

"   Group Key: p.city_id"

"   -> Sort (cost=84897.05..84897.05 rows=2 width=16) (actual time=10528.455..10528.590 rows=553 loops=1)"

"   Sort Key: p.city_id"

"   Sort Method: quicksort Memory: 50kB"

"   -> Nested Loop (cost=84879.20..84897.04 rows=2 width=16) (actual time=9609.156..10525.154 rows=553 loops=1)"

"   -> Nested Loop (cost=84878.77..84895.84 rows=2 width=28) (actual time=9609.053..10453.892 rows=553 loops=1)"

"   -> Finalize GroupAggregate (cost=84878.34..84878.93 rows=2 width=12) (actual time=9605.208..9773.332 rows=553 loops=1)"

"   Group Key: js.""StudentAMKA"""

"   Filter: (count(js.""ProgramID"") >= 2)"

"   Rows Removed by Filter: 58972"

"   -> Gather Merge (cost=84878.34..84878.83 rows=4 width=20) (actual time=9605.144..9731.956 rows=59527 loops=1)"

"   Workers Planned: 2"

"   Workers Launched: 2"

"   -> Partial GroupAggregate (cost=83878.31..83878.35 rows=2 width=20) (actual time=9485.487..9505.101 rows=19842 loops=3)"

"   Group Key: js.""StudentAMKA"""

"   -> Sort (cost=83878.31..83878.32 rows=2 width=16) (actual time=9483.488..9485.996 rows=20027 loops=3)"

"   Sort Key: js.""StudentAMKA"""

"   Sort Method: quicksort Memory: 1709kB"

"   Worker 0: Sort Method: quicksort Memory: 1681kB"

" Worker 1: Sort Method: quicksort Memory: 1731kB"

" -> Nested Loop (cost=130.70..83878.30 rows=2 width=16) (actual time=5.753..9444.430 rows=20027 loops=3)"

" -> Hash Join (cost=130.27..83836.21 rows=68 width=16) (actual time=4.431..1646.865 rows=180030 loops=3)"

" Hash Cond: (js.""ProgramID"" = pgr.""ProgramID"")"

" -> Parallel Index Only Scan using ""Joins_pkey"" on ""Joins"" js (cost=0.43..81021.51 rows=1022792 width=16) (actual time=3.716..1406.071 rows=821887 loops=3)"

" Heap Fetches: 500334"

" -> Hash (cost=129.43..129.43 rows=33 width=4) (actual time=0.462..0.463 rows=11 loops=3)"

" Buckets: 1024 Batches: 1 Memory Usage: 9kB"

" -> Bitmap Heap Scan on ""Program"" pgr (cost=4.26..129.43 rows=33 width=4) (actual time=0.439..0.447 rows=11 loops=3)"

" Recheck Cond: (""Duration"" = 5)"

" Heap Blocks: exact=2"

" -> Bitmap Index Scan on idx_program_duration (cost=0.00..4.25 rows=33 width=0) (actual time=0.421..0.421 rows=11 loops=3)"

" Index Cond: (""Duration"" = 5)"

" -> Index Scan using ""Student_pkey"" on ""Student"" (cost=0.43..0.62 rows=1 width=12) (actual time=0.042..0.042 rows=0 loops=540089)"

" Index Cond: ((amka)::text = (js.""StudentAMKA"")::text)"

" Filter: ((entry_date >= '2040-09-01'::date) AND (entry_date <= '2050-09-30'::date))"

" Rows Removed by Filter: 1"

" -> Index Scan using ""Person_pkey"" on ""Person"" p (cost=0.43..8.45 rows=1 width=16) (actual time=1.213..1.213 rows=1 loops=553)"

" Index Cond: ((amka)::text = (js.""StudentAMKA"")::text)"

" -> Index Only Scan using ""Student_pkey"" on ""Student"" s (cost=0.43..0.60 rows=1 width=12) (actual time=0.113..0.113 rows=1 loops=553)"

" Index Cond: (amka = (p.amka)::text)"

" Heap Fetches: 553"

"Planning Time: 3.014 ms"

"Execution Time: 10535.286 ms"