Μάθημα: «Συστήματα Διαχείρισης Βάσεων Δεδομένων (5ο εξ.)»

Ομάδα Εργασίας:

ΑΜ : Π15128, Σκάρος Γεώργιος
ΑΜ : Π15119, Ποδηματάς Δημήτριος
ΑΜ : Π15123, Ιωσήφ Σαγιέντ

Ημερομηνία παράδοσης : 31/01/2018

Μέρος 1ο

Πριν την εκτέλεση των ερωτημάτων του πρώτου μέρους έπρεπε να φτιάξουμε 1 πίνακα στην βάση δεδομένων που φτιάξαμε για να μπορέσουμε να του περάσουμε το dataset. Οι εντολές που χρησιμοποιήθηκαν για τη δημιουργία του πίνακα (Create table) και την εισαγωγή των δεδομένων σε αυτόν είναι τα εξής :

Δημιουργία πίνακα

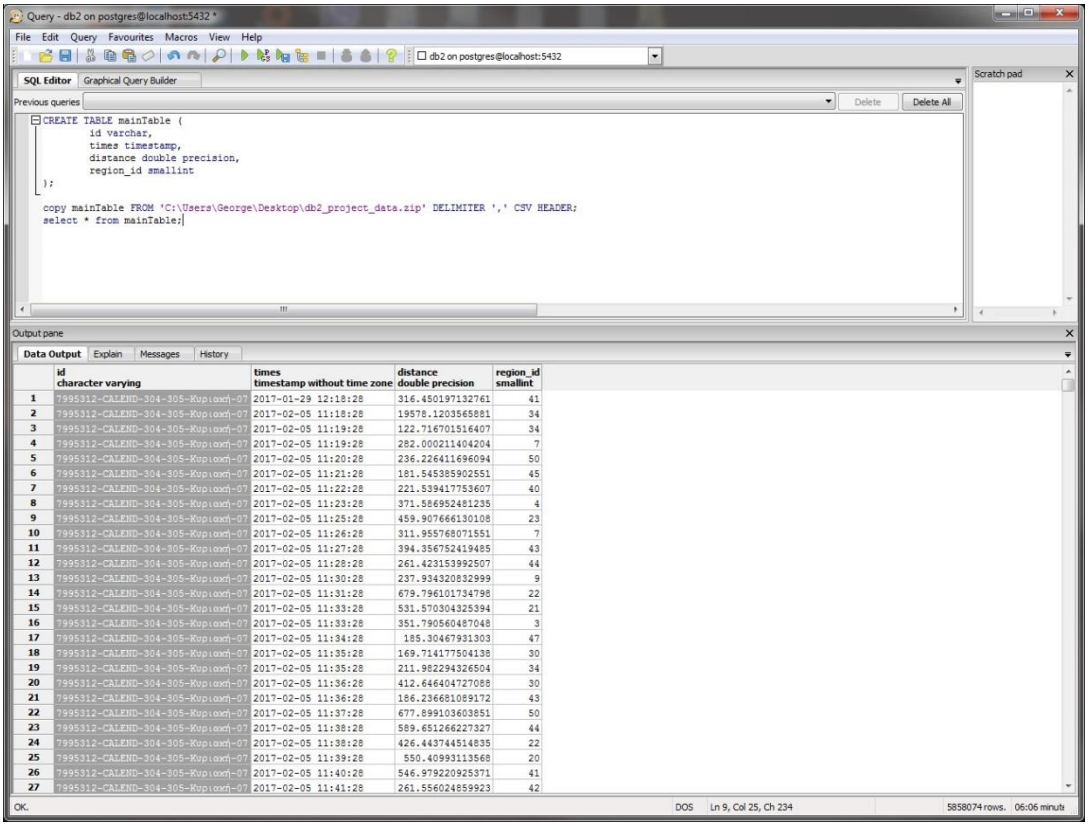
```
CREATE TABLE mainTable (
        id varchar,
        times timestamp,
        distance double precision,
        region_id smallint
);
```

Εισαγωγή δεδομένων

```
COPY mainTable(id, times, distance, region_id)
FROM 'C:\Desktop\db2_project_data.zip'
WITH CSV HEADER;
```

Ανανέωση στατιστικών

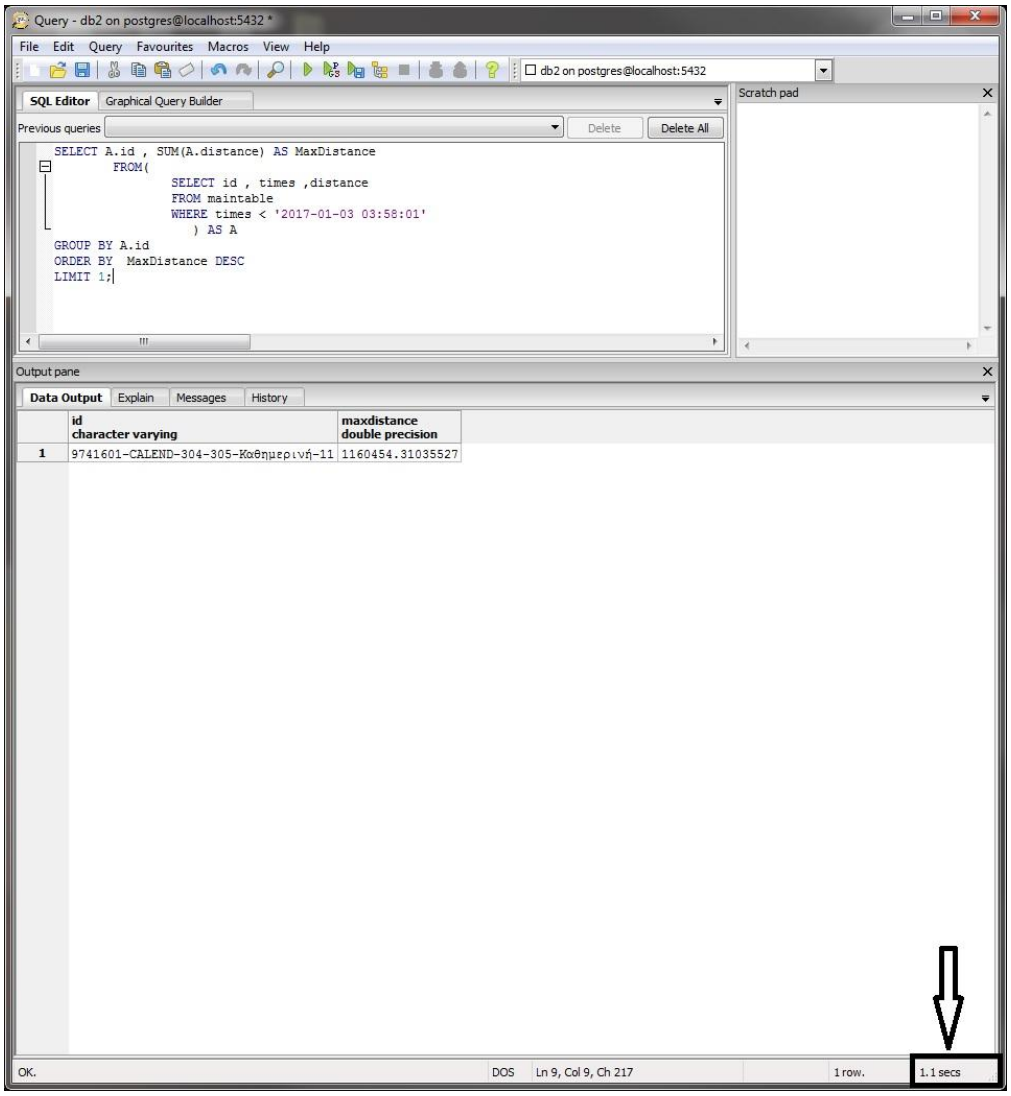
```
VACUUM FULL mainTable;
```



- Α. ερώτημα

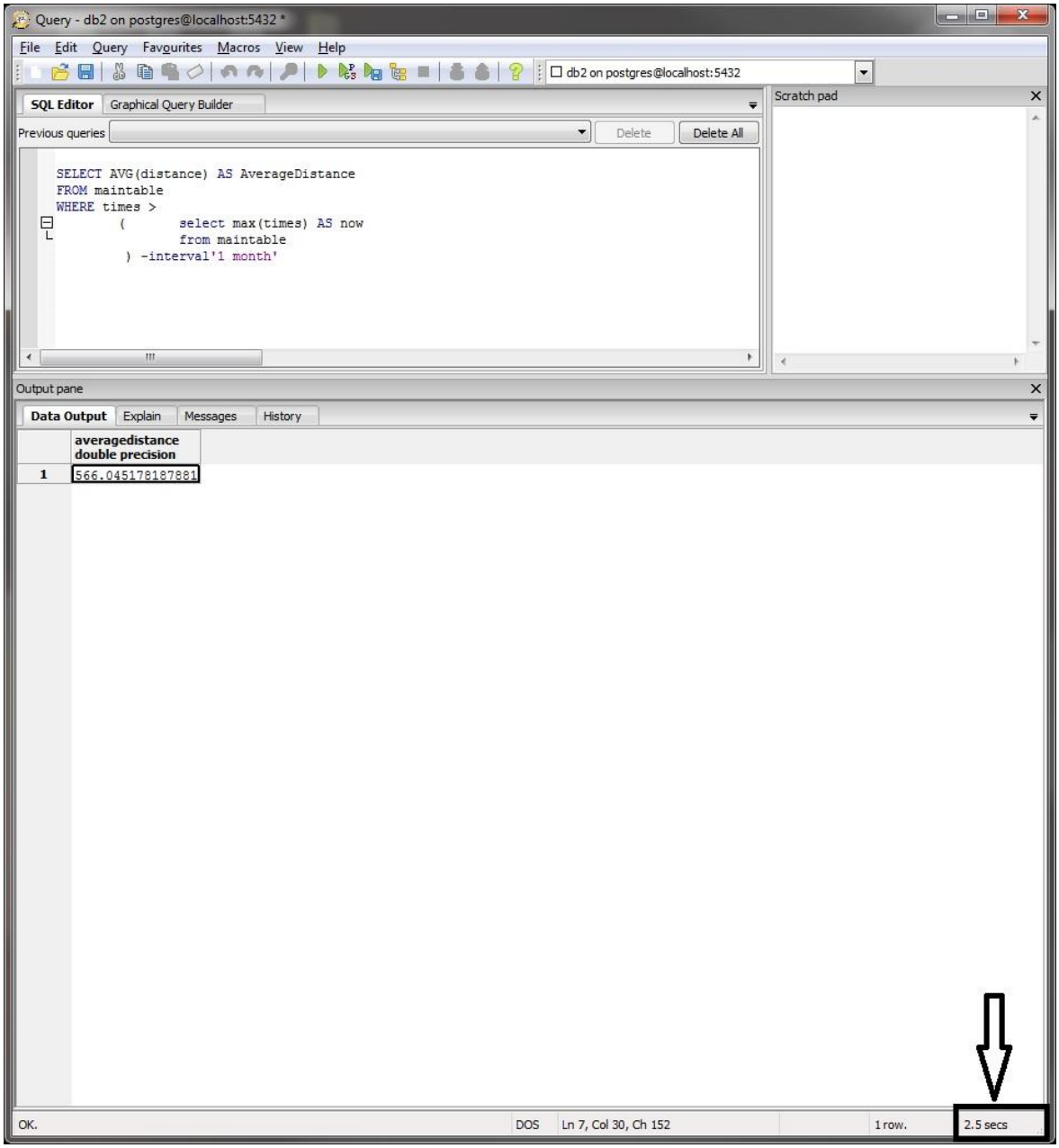
Τα queries του Α ερωτήματος είναι τα παρακάτω:

I.
```
SELECT A.id , SUM(A.distance) AS MaxDistance
FROM(
        SELECT id , times ,distance
        FROM maintable
        WHERE times < '2017-01-03 03:58:01'
       ) AS A
GROUP BY A.id
ORDER BY  MaxDistance DESC
LIMIT 1;
```

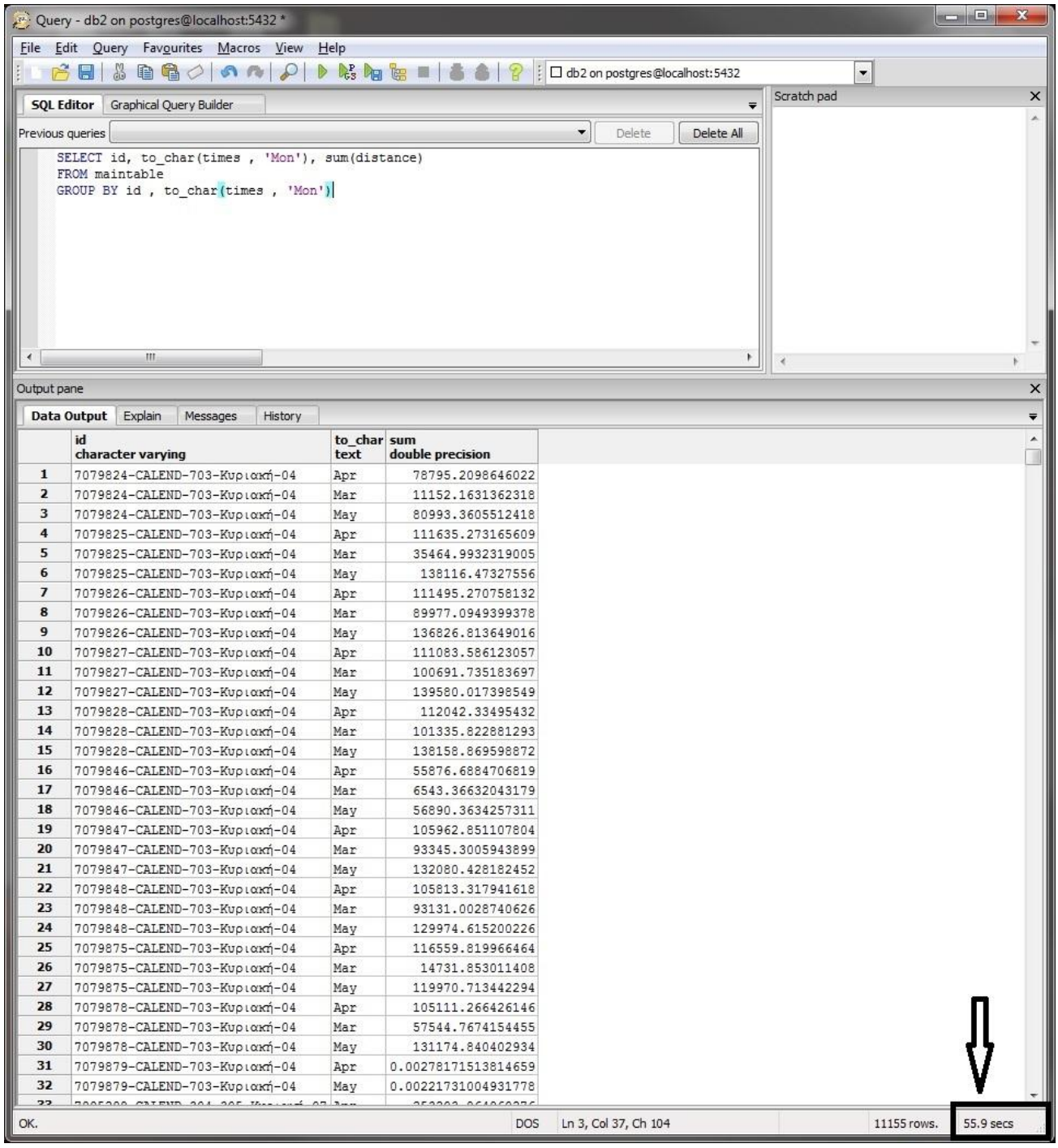


II.

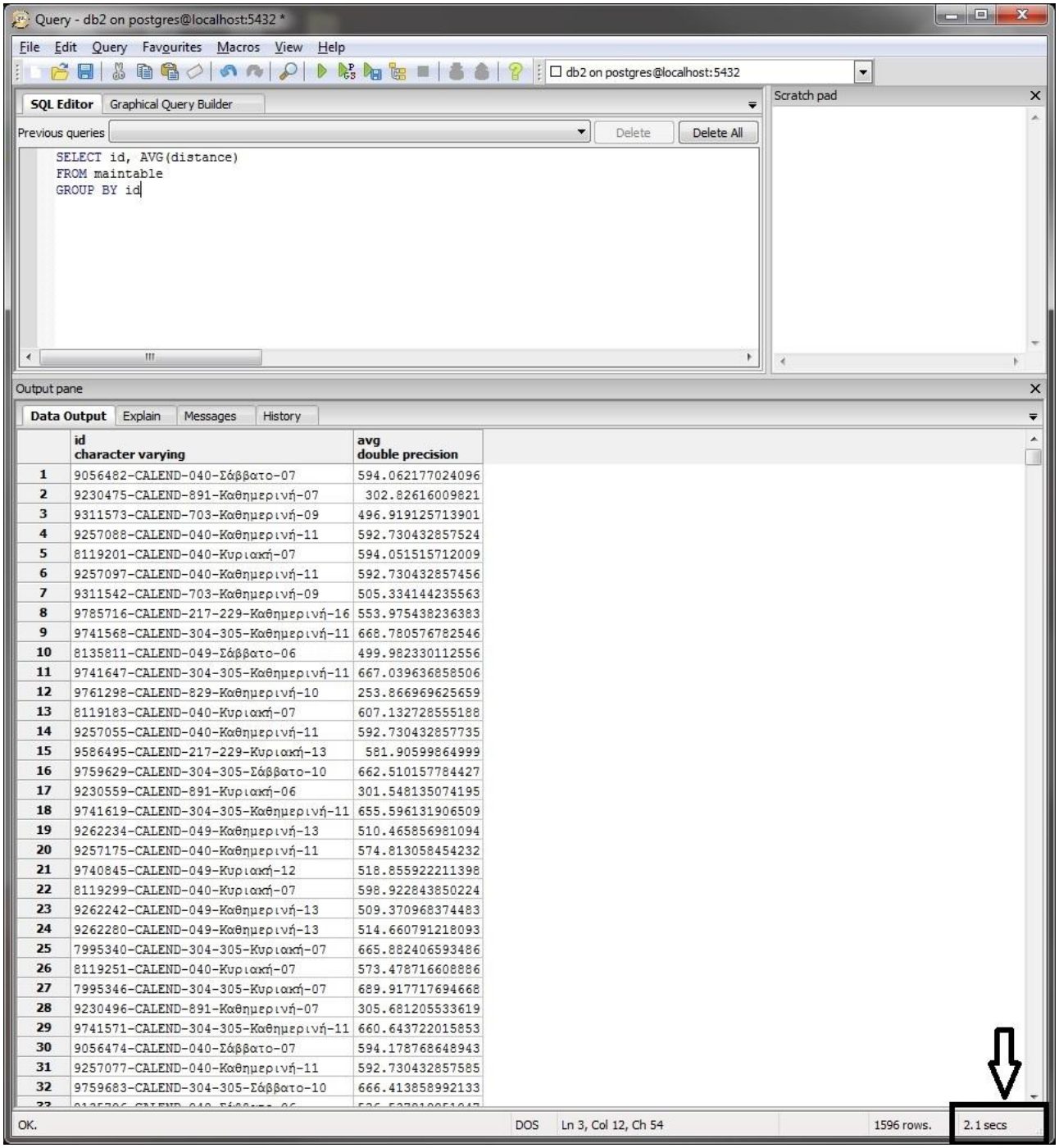
```
SELECT AVG(distance) AS AverageDistance
FROM maintable
WHERE times >
        (       select max(times) AS now
                from maintable
         ) -interval'1 month';
```



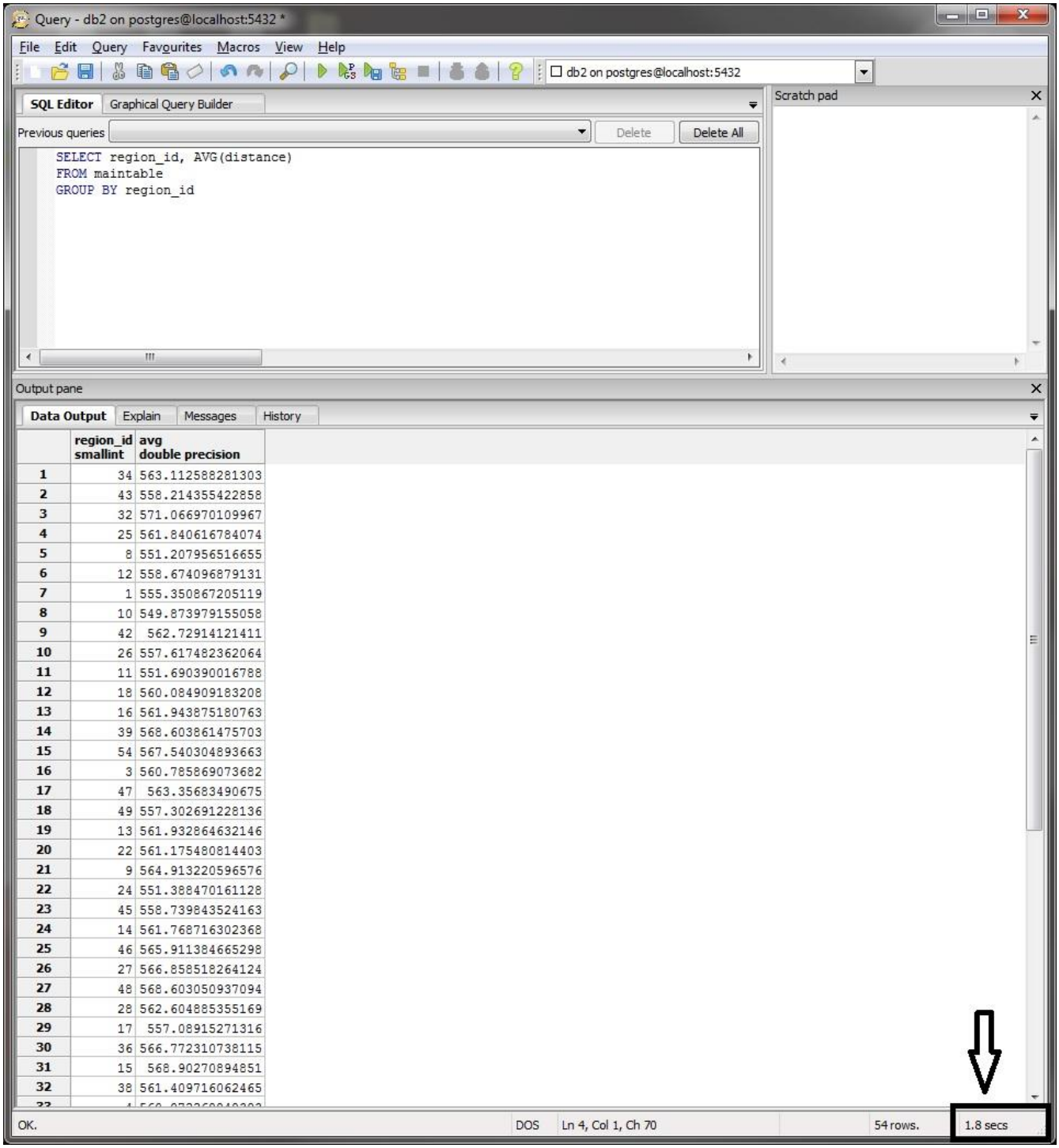
III. SELECT id, to_char(times , 'Mon'), sum(distance)
FROM maintable
GROUP BY id , to_char(times , 'Mon');



IV. SELECT id, AVG(distance)
FROM maintable
GROUP BY id;



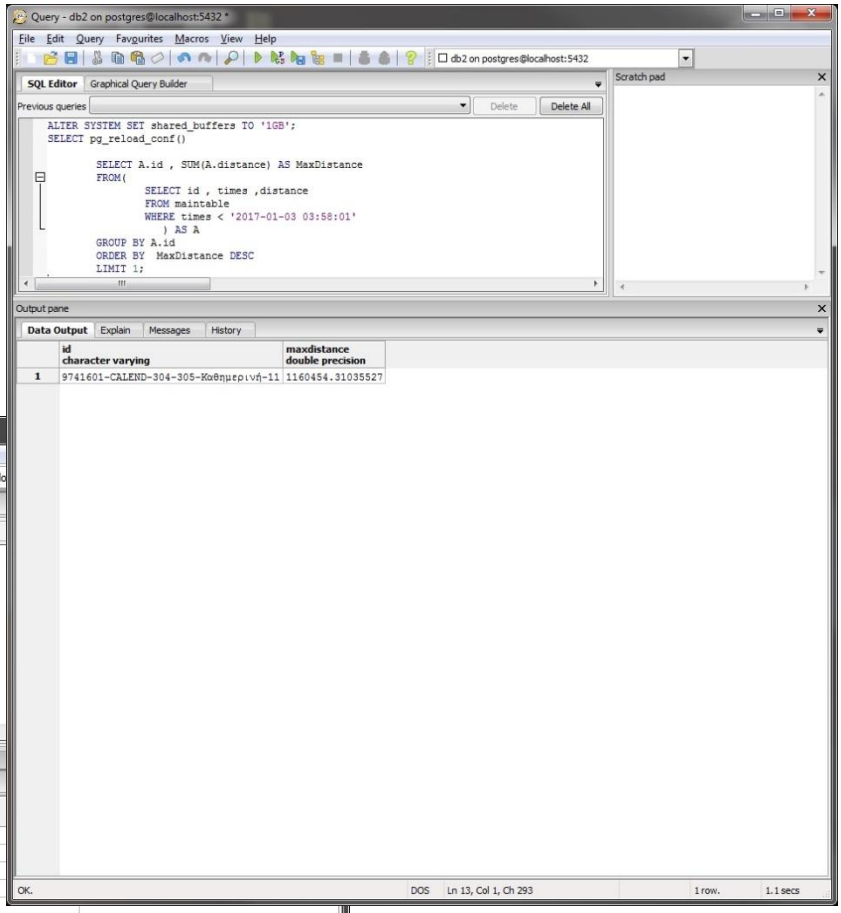
V. SELECT region_id, AVG(distance)
FROM maintable
GROUP BY region_id;

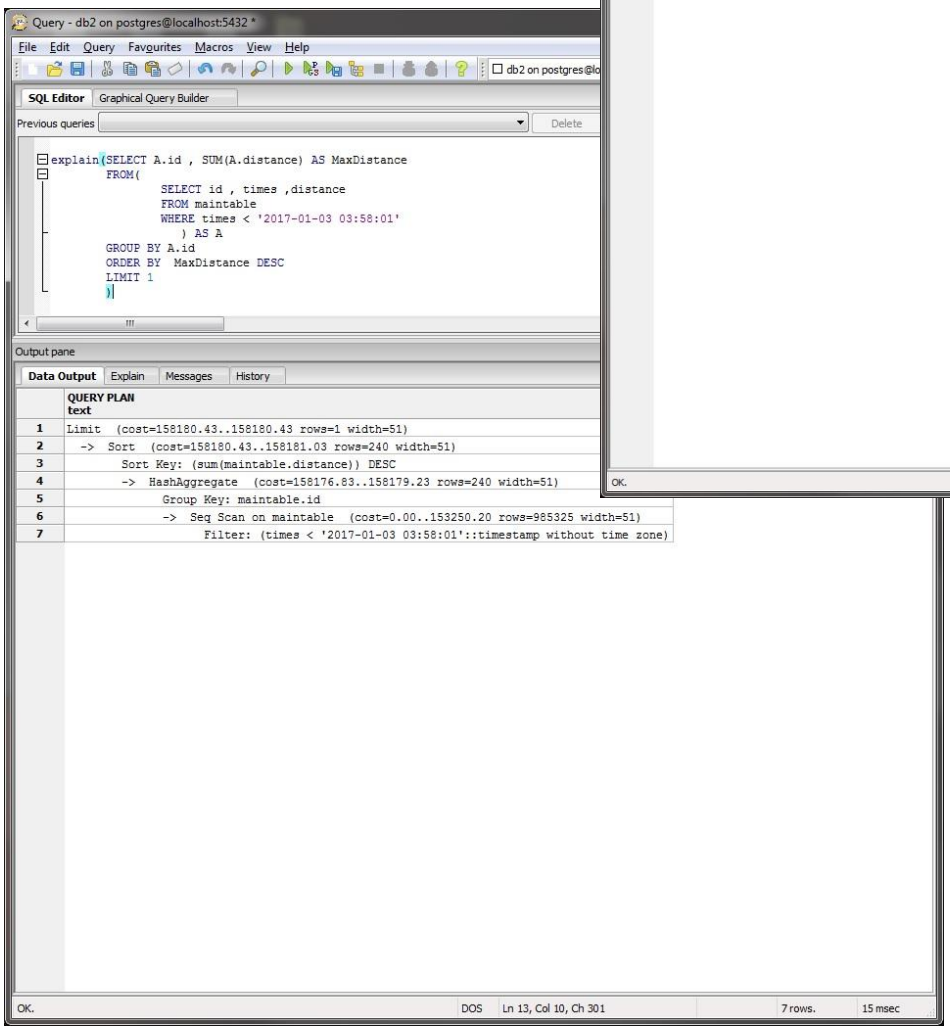


- Β. Ερώτημα

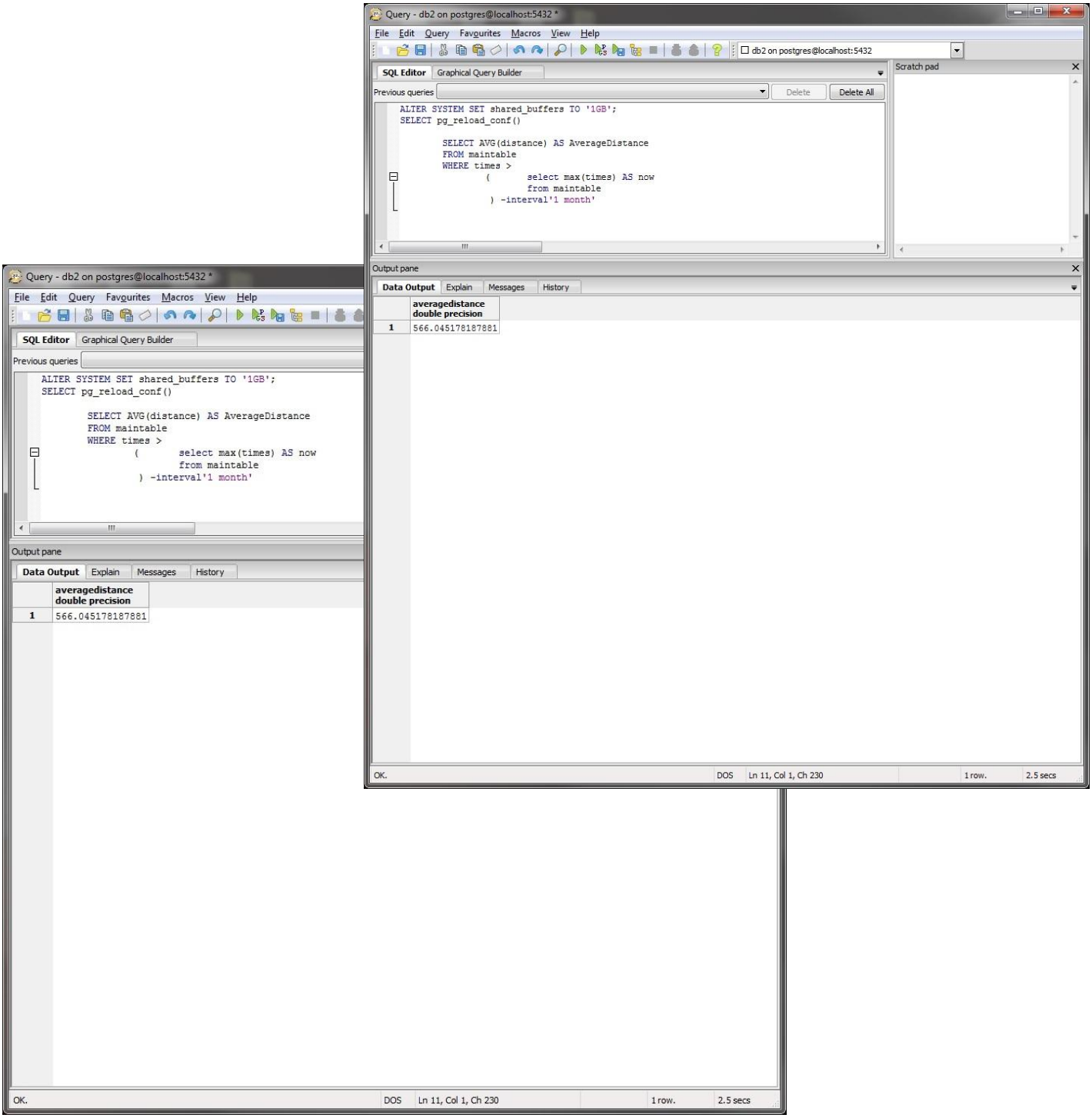
Για το β ερώτημα χρειάζεται αρχικά να δώσουμε περισσότερη μνήμη στους buffers με την εντολή ALTER SYSTEM SET shared_buffers TO '1GB';. Έπειτα χρειάζεται να κάνουμε επανεκκίνηση του postgreSQL server με την εντολή SELECT pg_reload_conf();. Στην συνέχεια εκτέλεσα τα queries του προηγούμενου ερωτήματος. Σε αυτό το σημείο είναι σημαντικό να επισημάνω ότι στον υπολογιστή μου, στο σπίτι όπου εκτελούσα τα queries έχω ssd οπότε σε αυτό το ερώτημα δεν θα δούμε μεγάλες διάφορες στους χρόνους. Παρόλα αυτά έχω συμπεριλάβει στα screenshot που έχω κάνει και αυτά της κανονικής εκτέλεσης αλλά και εκείνα του explain.

i.

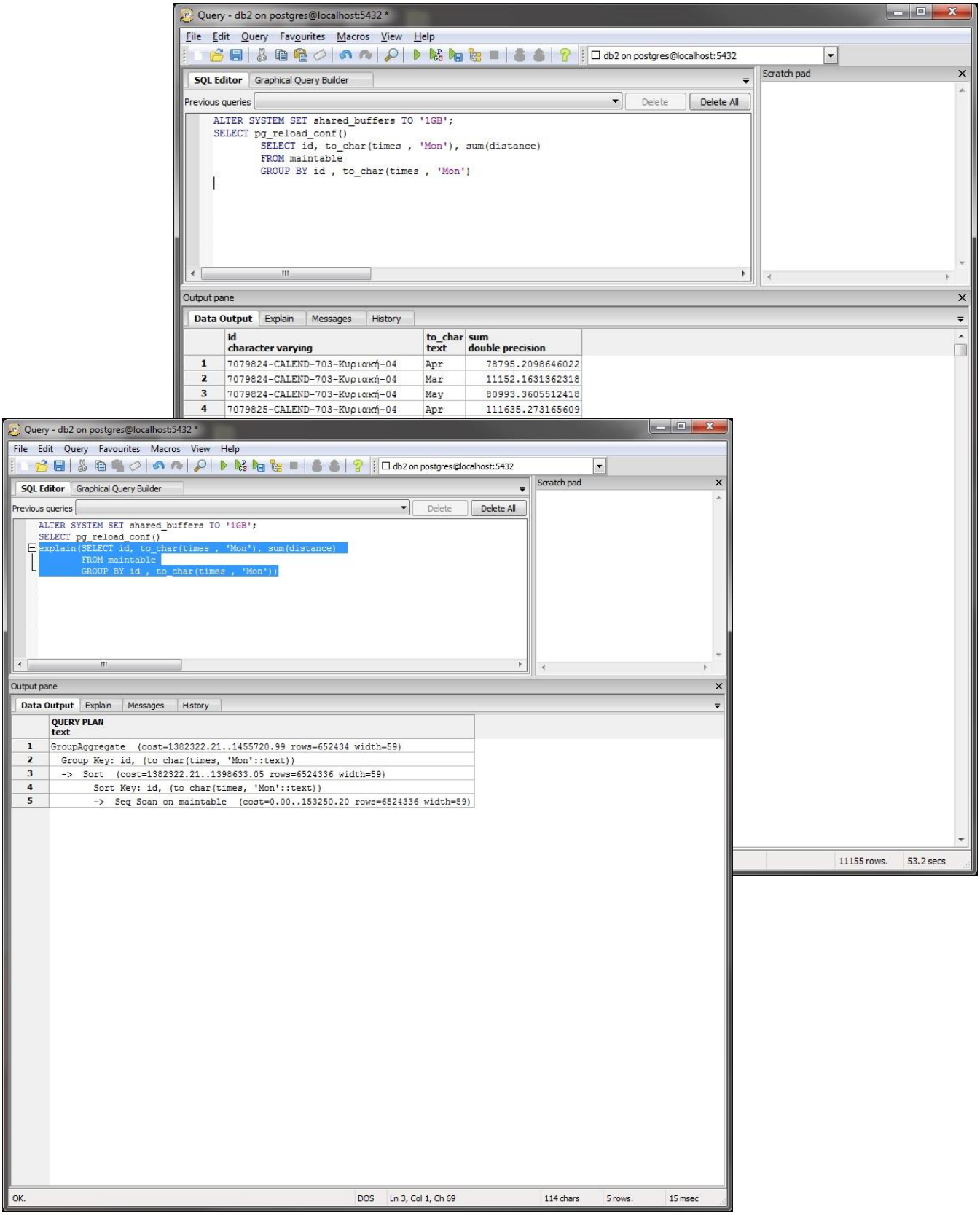




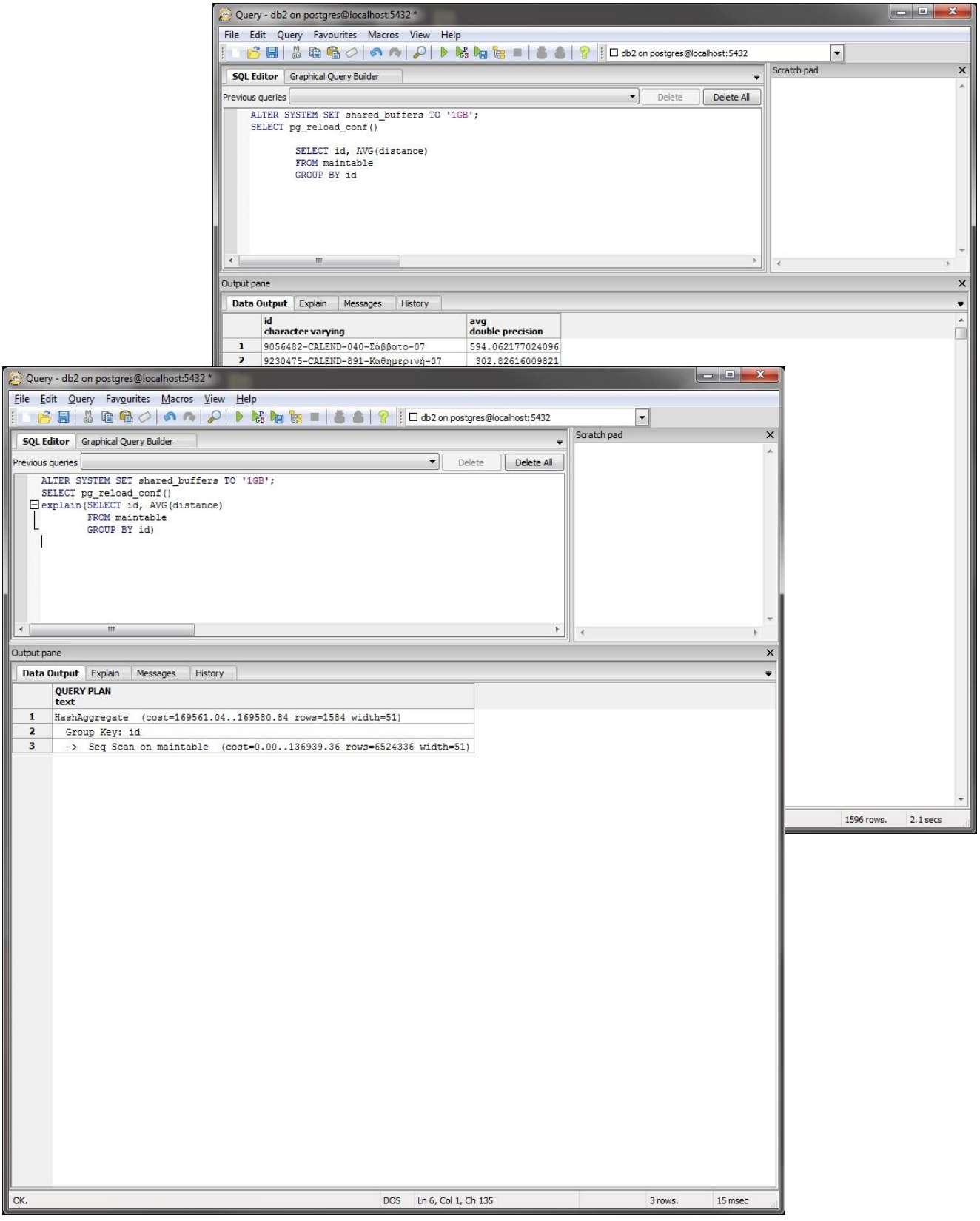
ii.


Query - db2 on postgres@localhost:5432 *
File Edit Query Favourites Macros View Help
db2 on postgres@localhost:5432
SQL Editor
Graphical Query Builder
Previous queries
Delete
Delete All
Scratch pad
ALTER SYSTEM SET shared_buffers TO '1GB';
SELECT pg_reload_conf()
SELECT AVG(distance) AS AverageDistance
FROM maintable
WHERE times >
(select max(times) AS now
from maintable
) -interval'1 month'
Output pane
Data Output
Explain
Messages
History
averagedistance
double precision
1
566.045178187881
OK.
DOS
Ln 11, Col 1, Ch 230
1 row.
2.5 secs

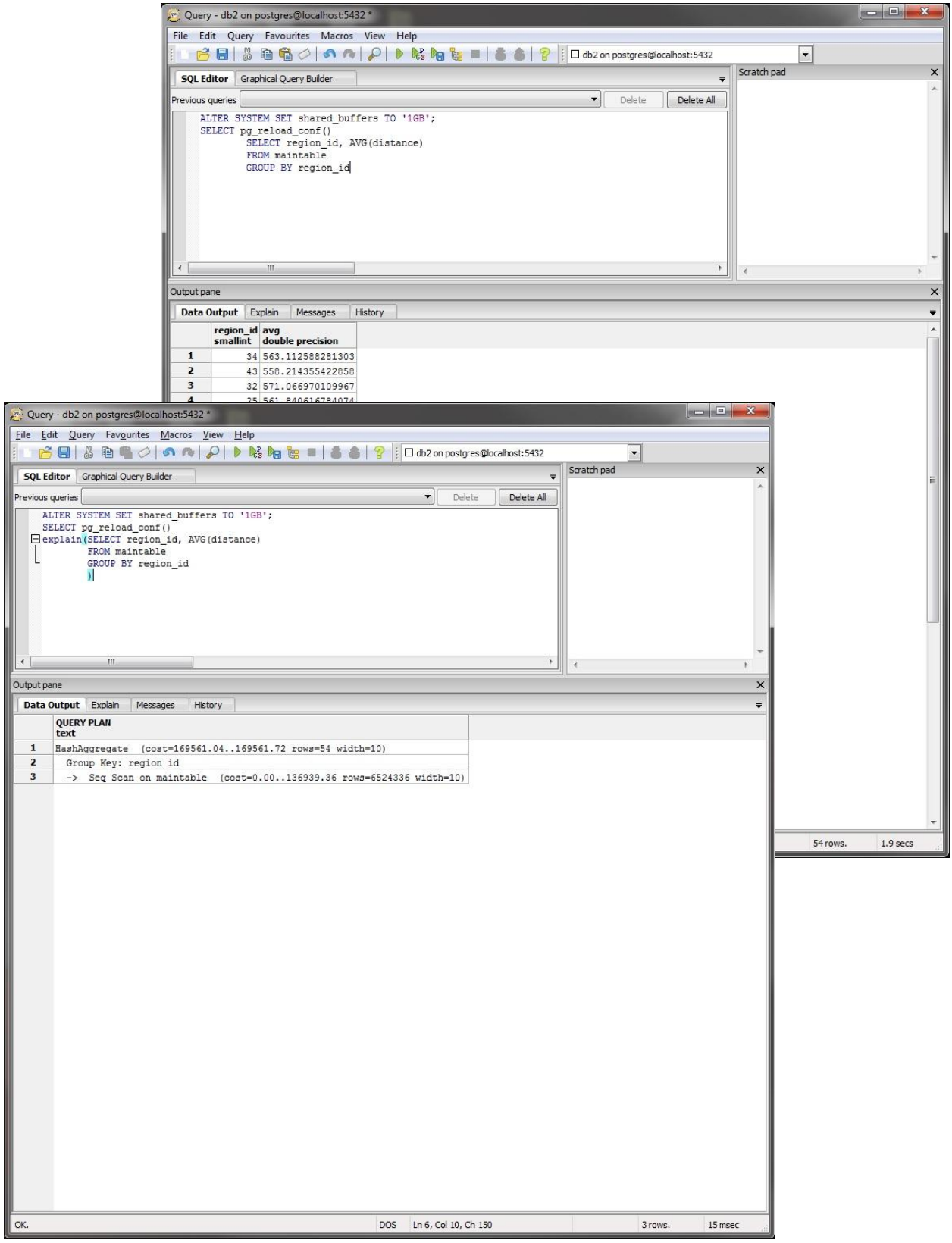
iii.


Query - db2 on postgres@localhost:5432 *
File Edit Query Favourites Macros View Help
db2 on postgres@localhost:5432
SQL Editor
Graphical Query Builder
Previous queries
Delete
Delete All
Scratch pad
ALTER SYSTEM SET shared_buffers TO '1GB';
SELECT pg_reload_conf()
SELECT id, to_char(times , 'Mon'), sum(distance)
FROM maintable
GROUP BY id , to_char(times , 'Mon')
Output pane
Data Output
Explain
Messages
History
id
character varying
to_char
text
sum
double precision
1 7079824-CALEND-703-Κυριακή-04 Apr 78795.2098646022
2 7079824-CALEND-703-Κυριακή-04 Mar 11152.1631362318
3 7079824-CALEND-703-Κυριακή-04 May 80993.3605512418
4 7079825-CALEND-703-Κυριακή-04 Apr 111635.273165609
11155 rows.
53.2 secs
Query - db2 on postgres@localhost:5432 *
File Edit Query Favourites Macros View Help
db2 on postgres@localhost:5432
SQL Editor
Graphical Query Builder
Previous queries
Delete
Delete All
Scratch pad
ALTER SYSTEM SET shared_buffers TO '1GB';
SELECT pg_reload_conf()
explain(SELECT id, to_char(times , 'Mon'), sum(distance)
FROM maintable
GROUP BY id , to_char(times , 'Mon'))
Output pane
Data Output
Explain
Messages
History
QUERY PLAN
text
1 GroupAggregate (cost=1382322.21..1455720.99 rows=652434 width=59)
2 Group Key: id, (to_char(times, 'Mon'::text))
3 -> Sort (cost=1382322.21..1398633.05 rows=6524336 width=59)
4 Sort Key: id, (to_char(times, 'Mon'::text))
5 -> Seq Scan on maintable (cost=0.00..153250.20 rows=6524336 width=59)
OK.
DOS
Ln 3, Col 1, Ch 69
114 chars
5 rows.
15 msec

iv.

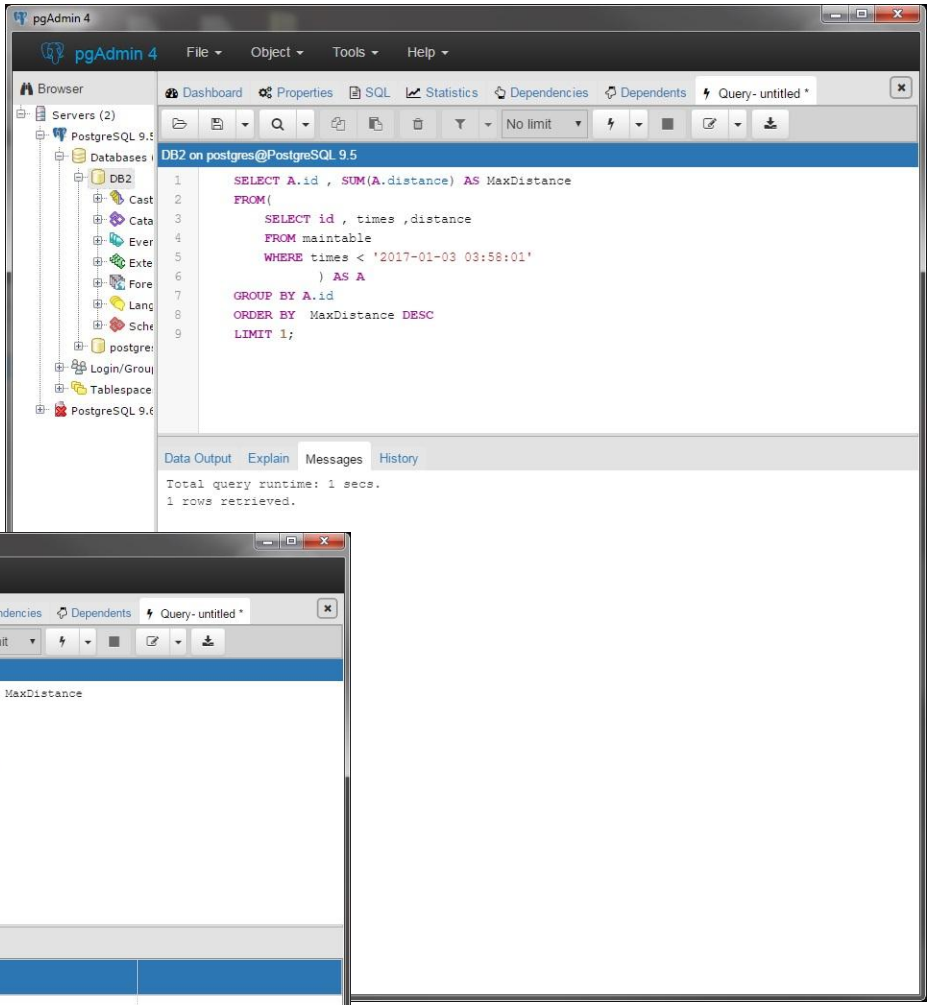

Query - db2 on postgres@localhost:5432 *
File Edit Query Favourites Macros View Help
db2 on postgres@localhost:5432
SQL Editor
Graphical Query Builder
Previous queries
Delete
Delete All
Scratch pad
ALTER SYSTEM SET shared_buffers TO '1GB';
SELECT pg_reload_conf()
SELECT id, AVG(distance)
FROM maintable
GROUP BY id
Output pane
Data Output
Explain
Messages
History
id
character varying
avg
double precision
1 9056482-CALEND-040-Σάββατο-07 594.062177024096
2 9230475-CALEND-891-Καθημερινή-07 302.82616009821
1596 rows.
2.1 secs
Query - db2 on postgres@localhost:5432 *
File Edit Query Favourites Macros View Help
db2 on postgres@localhost:5432
SQL Editor
Graphical Query Builder
Previous queries
Delete
Delete All
Scratch pad
ALTER SYSTEM SET shared_buffers TO '1GB';
SELECT pg_reload_conf()
explain(SELECT id, AVG(distance)
FROM maintable
GROUP BY id)
Output pane
Data Output
Explain
Messages
History
QUERY PLAN
text
1 HashAggregate (cost=169561.04..169580.84 rows=1584 width=51)
2 Group Key: id
3 -> Seq Scan on maintable (cost=0.00..136939.36 rows=6524336 width=51)
OK.
DOS
Ln 6, Col 1, Ch 135
3 rows.
15 msec

v.

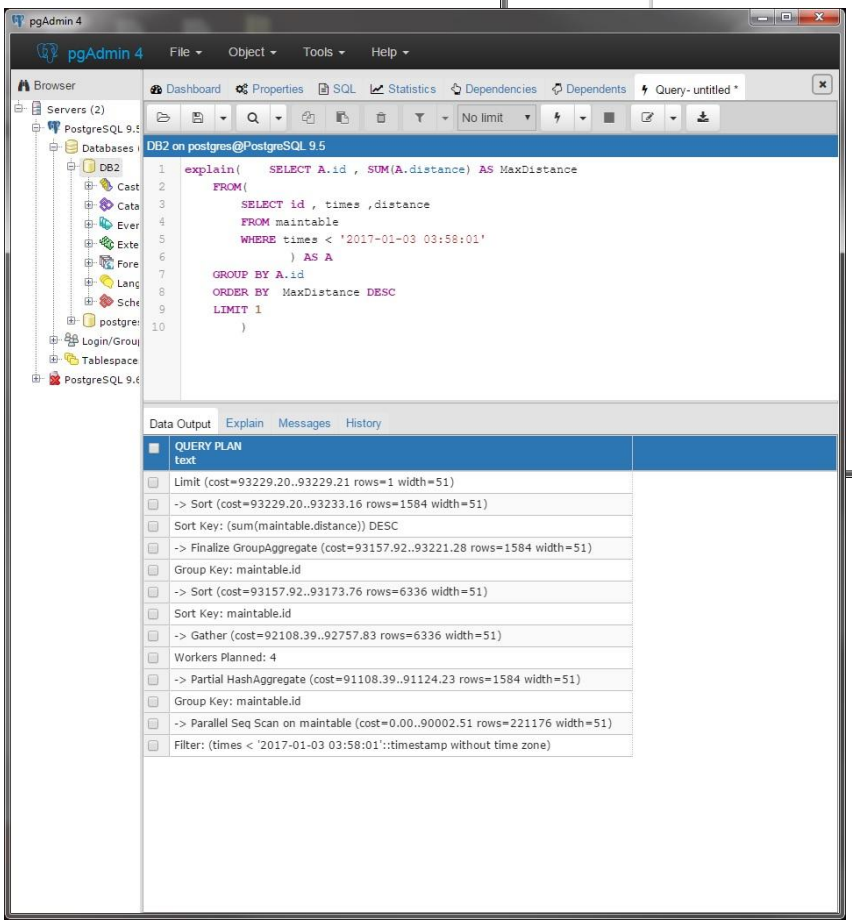

Query - db2 on postgres@localhost:5432 *
File Edit Query Favourites Macros View Help
db2 on postgres@localhost:5432
SQL Editor
Graphical Query Builder
Previous queries
Delete
Delete All
Scratch pad
ALTER SYSTEM SET shared_buffers TO '1GB';
SELECT pg_reload_conf()
SELECT region_id, AVG(distance)
FROM maintable
GROUP BY region_id
Output pane
Data Output
Explain
Messages
History
region_id smallint
avg double precision
1 34 563.112588281303
2 43 558.214355422858
3 32 571.066970109967
4 25 561.840616784074
Query - db2 on postgres@localhost:5432 *
File Edit Query Favourites Macros View Help
db2 on postgres@localhost:5432
SQL Editor
Graphical Query Builder
Previous queries
Delete
Delete All
Scratch pad
ALTER SYSTEM SET shared_buffers TO '1GB';
SELECT pg_reload_conf()
explain(SELECT region_id, AVG(distance)
FROM maintable
GROUP BY region_id
)
Output pane
Data Output
Explain
Messages
History
QUERY PLAN text
1 HashAggregate (cost=169561.04..169561.72 rows=54 width=10)
2 Group Key: region_id
3 -> Seq Scan on maintable (cost=0.00..136939.36 rows=6524336 width=10)
OK.
DOS
Ln 6, Col 10, Ch 150
3 rows.
15 msec
54 rows.
1.9 secs

- Γ. Ερώτημα

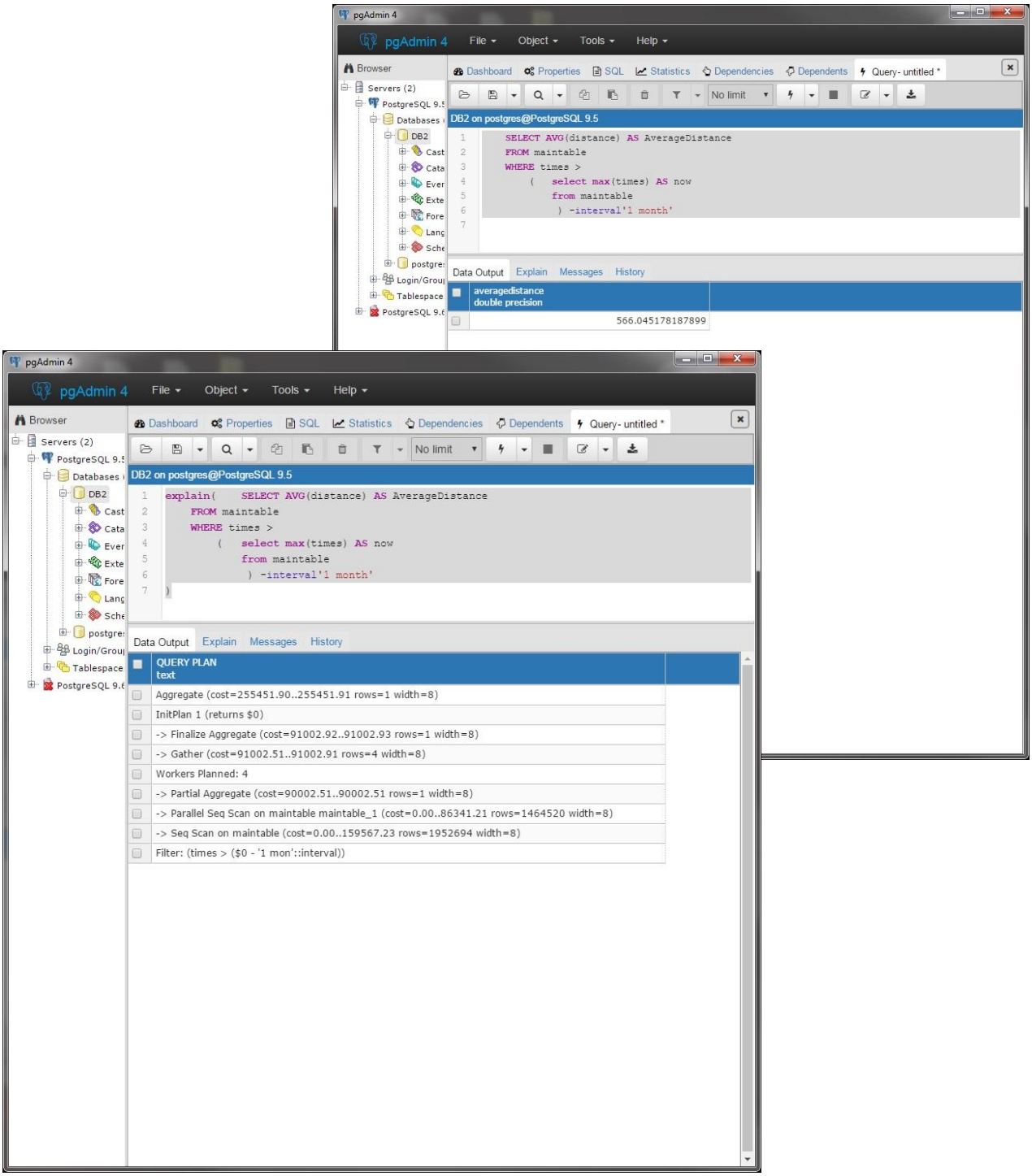
Στο 3ο ερώτημα της εργασίας χρειάστηκε, πριν πράξω οτιδήποτε , να εγκαταστήσω νεότερη έκδοση της postgreSQL διότι δεν υπήρχε η εντολή max_parallel_workers_per_gather ή κάποια άλλη παρόμοια της στην έκδοση 9.5. Αυτός είναι και ο λόγος που στα screenshot έχει αλλάξει και το ui από pgadmin3 σε pgadmin4. Αφού έκανα την αλλαγή, εκτέλεσα την εντολή ALTER SYSTEM SET max_parallel_workers_per_gather = 30; και μετά έκανα επανεκκίνηση του server. Σε αυτό το ερώτημα παρατηρούνται αλλαγές διότι αυτή η εντολή αλλάζει την χρήση του επεξεργαστή που γίνεται από την postgreSQL. Η μεγαλύτερη διαφορά παρατηρείται στο 3ο ερώτημα διότι αυτό είχε τον μεγαλύτερο χρόνο εκτέλεσης. Στα Screenshot που θα δείτε παρακάτω έκρινα καλύτερο να φαίνεται ο χρόνος εκτέλεσης και όχι το αποτέλεσμα από εκείνα έχουν φανεί σε προηγούμενα screenshot. Αυτά είναι τα αποτελέσματα που είδα:



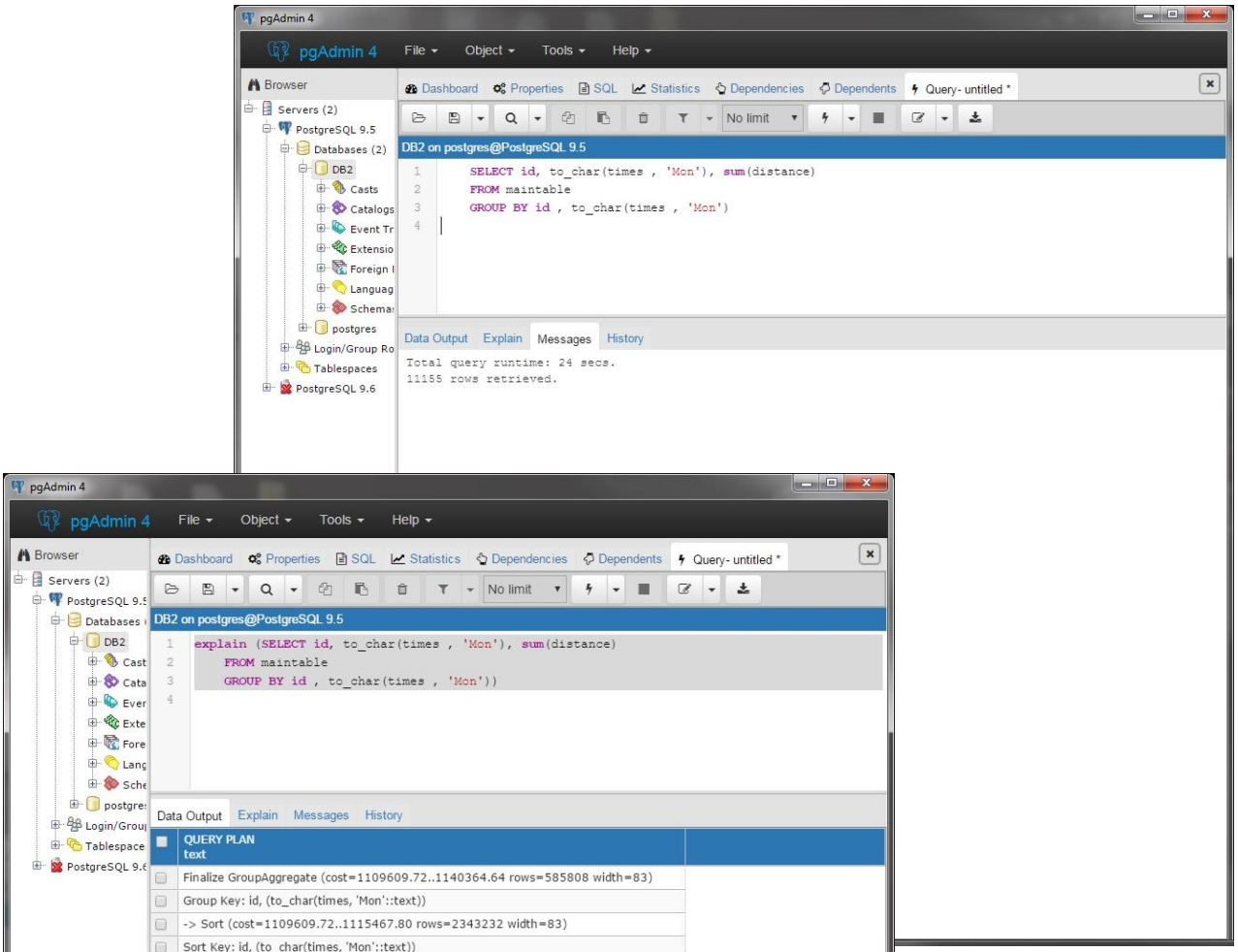
i.

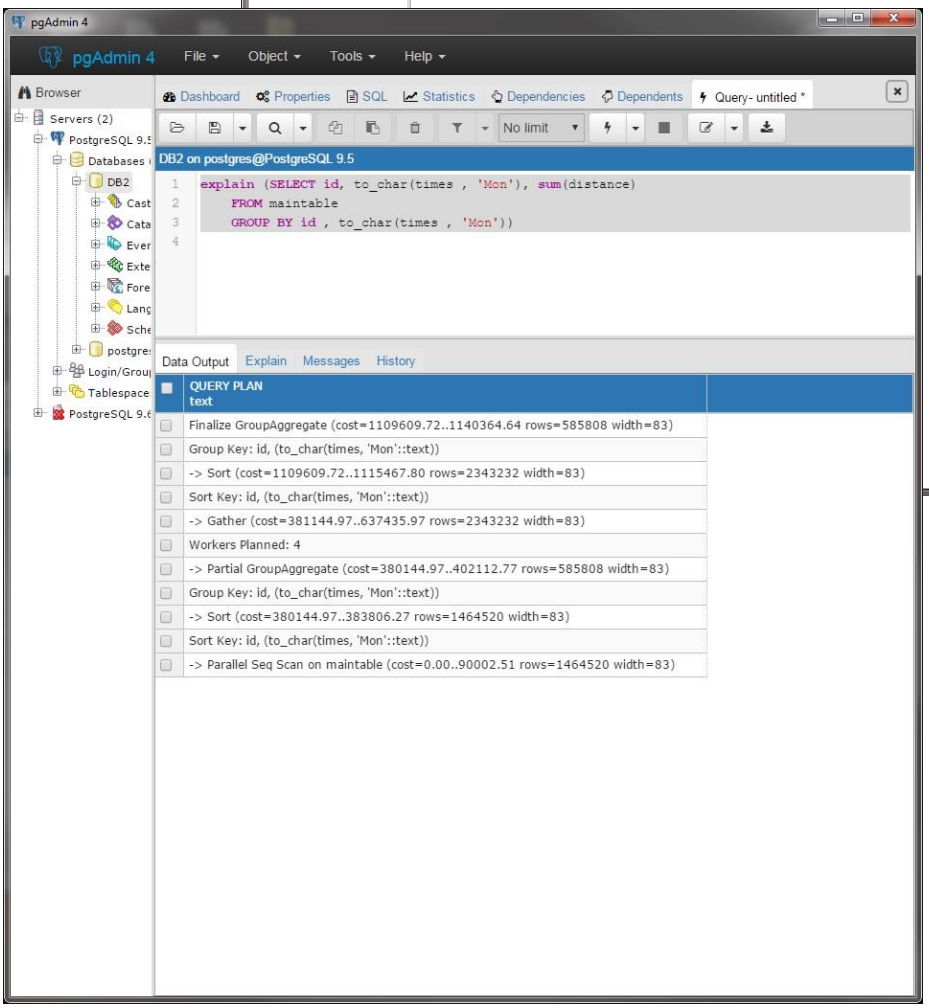


ii.

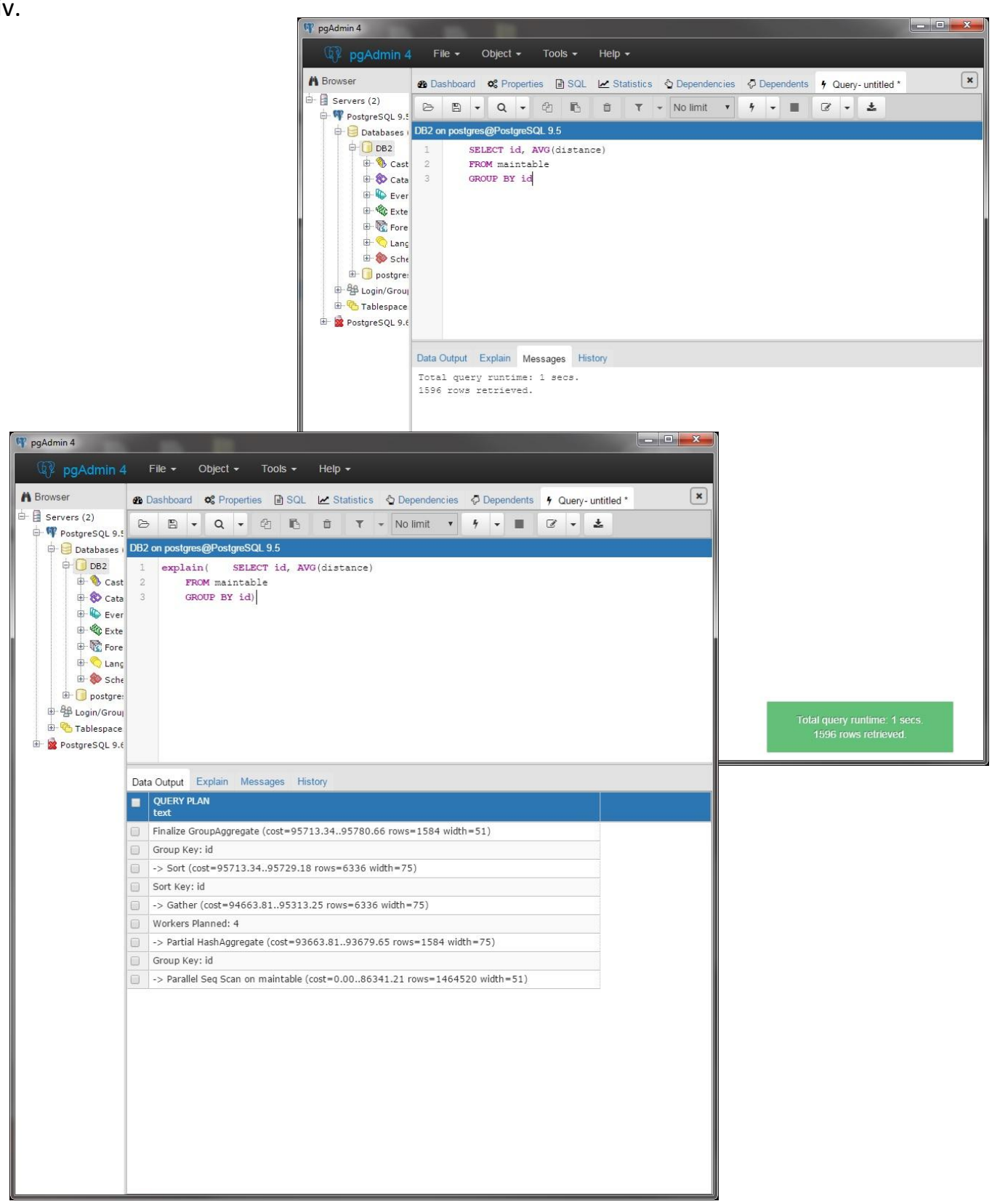

pgAdmin 4
File
Object
Tools
Help
Browser
Servers (2)
Dashboard
Properties
SQL
Statistics
Dependencies
Dependents
Query- untitled *
No limit
DB2 on postgres@PostgreSQL 9.5
SELECT AVG(distance) AS AverageDistance
FROM maintable
WHERE times >
(select max(times) AS now
from maintable
) -interval'1 month'
Data Output
Explain
Messages
History
averagedistance
double precision
566.045178187899
explain(SELECT AVG(distance) AS AverageDistance
FROM maintable
WHERE times >
(select max(times) AS now
from maintable
) -interval'1 month'
)
QUERY PLAN
text
Aggregate (cost=255451.90..255451.91 rows=1 width=8)
InitPlan 1 (returns $0)
-> Finalize Aggregate (cost=91002.92..91002.93 rows=1 width=8)
-> Gather (cost=91002.51..91002.91 rows=4 width=8)
Workers Planned: 4
-> Partial Aggregate (cost=90002.51..90002.51 rows=1 width=8)
-> Parallel Seq Scan on maintable maintable_1 (cost=0.00..86341.21 rows=1464520 width=8)
-> Seq Scan on maintable (cost=0.00..159567.23 rows=1952694 width=8)
Filter: (times > ($0 - '1 mon'::interval))

iii.

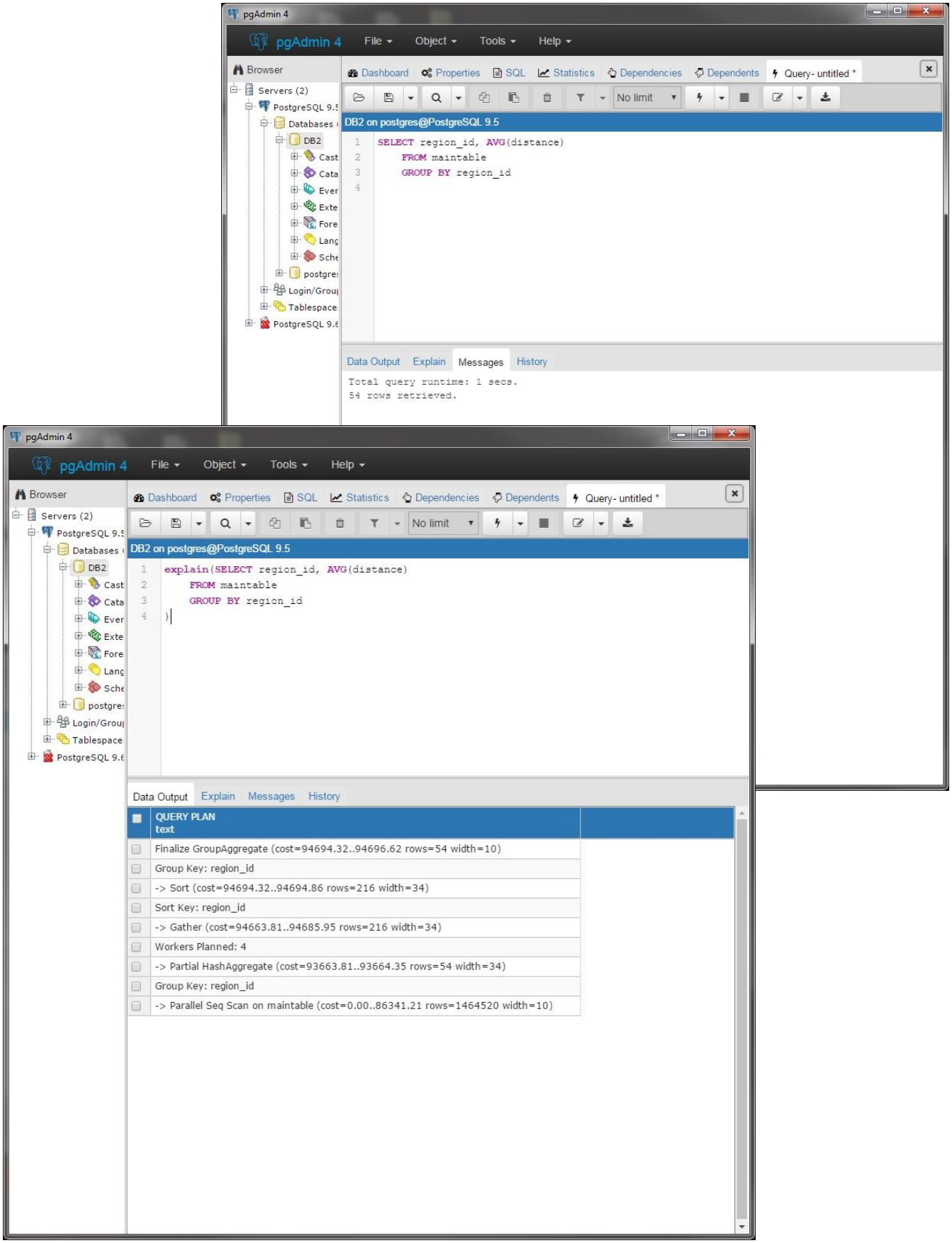

pgAdmin 4
File
Object
Tools
Help
Browser
Dashboard
Properties
SQL
Statistics
Dependencies
Dependents
Query- untitled *
No limit
DB2 on postgres@PostgreSQL 9.5
SELECT id, to_char(times , 'Mon'), sum(distance)
FROM maintable
GROUP BY id , to_char(times , 'Mon')
Data Output
Explain
Messages
History
Total query runtime: 24 secs.
11155 rows retrieved.


pgAdmin 4
File
Object
Tools
Help
Browser
Dashboard
Properties
SQL
Statistics
Dependencies
Dependents
Query- untitled *
No limit
DB2 on postgres@PostgreSQL 9.5
explain (SELECT id, to_char(times , 'Mon'), sum(distance)
FROM maintable
GROUP BY id , to_char(times , 'Mon'))
Data Output
Explain
Messages
History
QUERY PLAN
text
Finalize GroupAggregate (cost=1109609.72..1140364.64 rows=585808 width=83)
Group Key: id, (to_char(times, 'Mon'::text))
-> Sort (cost=1109609.72..1115467.80 rows=2343232 width=83)
Sort Key: id, (to_char(times, 'Mon'::text))
-> Gather (cost=381144.97..637435.97 rows=2343232 width=83)
Workers Planned: 4
-> Partial GroupAggregate (cost=380144.97..402112.77 rows=585808 width=83)
Group Key: id, (to_char(times, 'Mon'::text))
-> Sort (cost=380144.97..383806.27 rows=1464520 width=83)
Sort Key: id, (to_char(times, 'Mon'::text))
-> Parallel Seq Scan on maintable (cost=0.00..90002.51 rows=1464520 width=83)

iv.


pgAdmin 4
File
Object
Tools
Help
Browser
Dashboard
Properties
SQL
Statistics
Dependencies
Dependents
Query- untitled *
No limit
DB2 on postgres@PostgreSQL 9.5
SELECT id, AVG(distance)
FROM maintable
GROUP BY id
Data Output
Explain
Messages
History
Total query runtime: 1 secs.
1596 rows retrieved.
explain(SELECT id, AVG(distance)
FROM maintable
GROUP BY id)
QUERY PLAN
text
Finalize GroupAggregate (cost=95713.34..95780.66 rows=1584 width=51)
Group Key: id
-> Sort (cost=95713.34..95729.18 rows=6336 width=75)
Sort Key: id
-> Gather (cost=94663.81..95313.25 rows=6336 width=75)
Workers Planned: 4
-> Partial HashAggregate (cost=93663.81..93679.65 rows=1584 width=75)
Group Key: id
-> Parallel Seq Scan on maintable (cost=0.00..86341.21 rows=1464520 width=51)

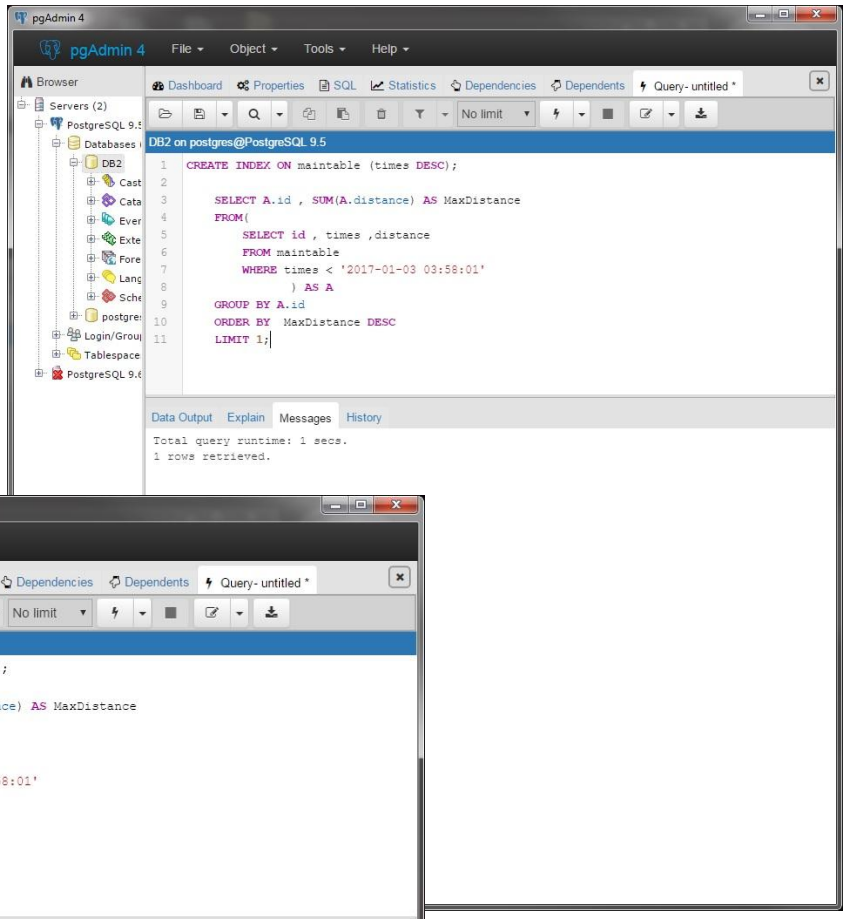
v.

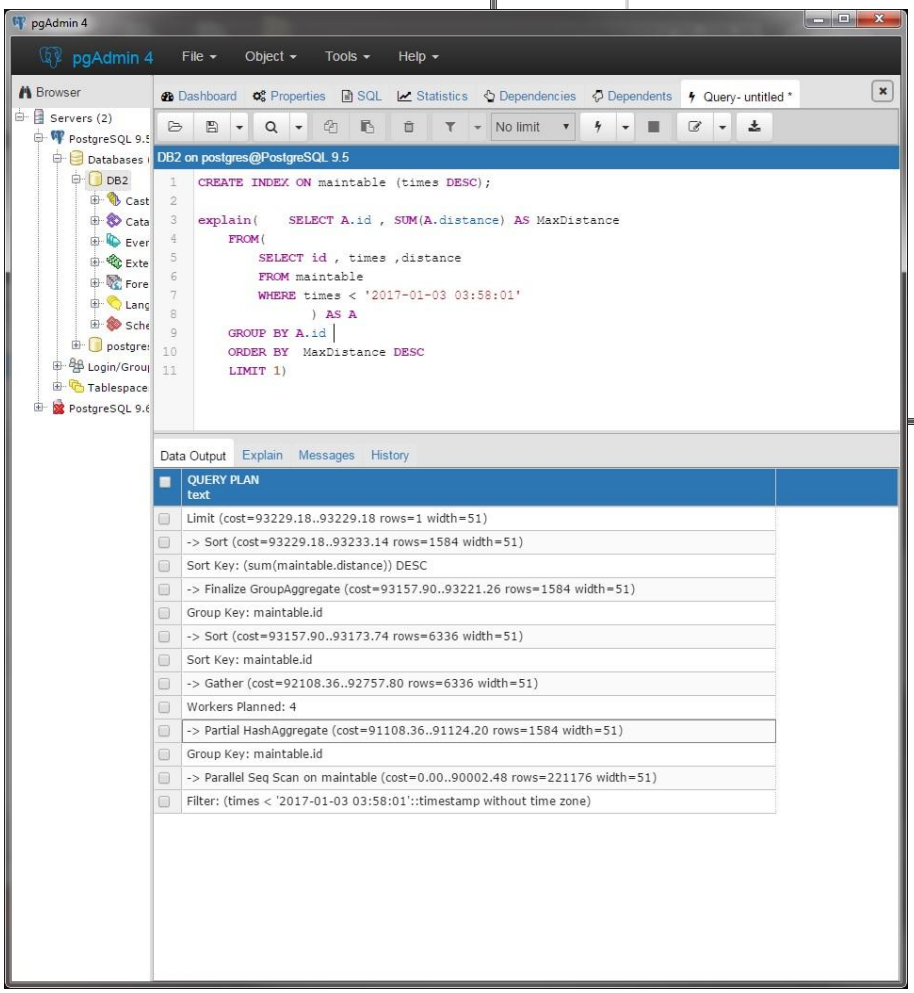

pgAdmin 4
File
Object
Tools
Help
Browser
Servers (2)
Dashboard
Properties
SQL
Statistics
Dependencies
Dependents
Query- untitled *
No limit
DB2 on postgres@PostgreSQL 9.5
SELECT region_id, AVG(distance)
FROM maintable
GROUP BY region_id
Data Output
Explain
Messages
History
Total query runtime: 1 secs.
54 rows retrieved.
explain(SELECT region_id, AVG(distance)
FROM maintable
GROUP BY region_id
)
QUERY PLAN
text
Finalize GroupAggregate (cost=94694.32..94696.62 rows=54 width=10)
Group Key: region_id
-> Sort (cost=94694.32..94694.86 rows=216 width=34)
Sort Key: region_id
-> Gather (cost=94663.81..94685.95 rows=216 width=34)
Workers Planned: 4
-> Partial HashAggregate (cost=93663.81..93664.35 rows=54 width=34)
Group Key: region_id
-> Parallel Seq Scan on maintable (cost=0.00..86341.21 rows=1464520 width=10)

- Δ. Ερώτημα

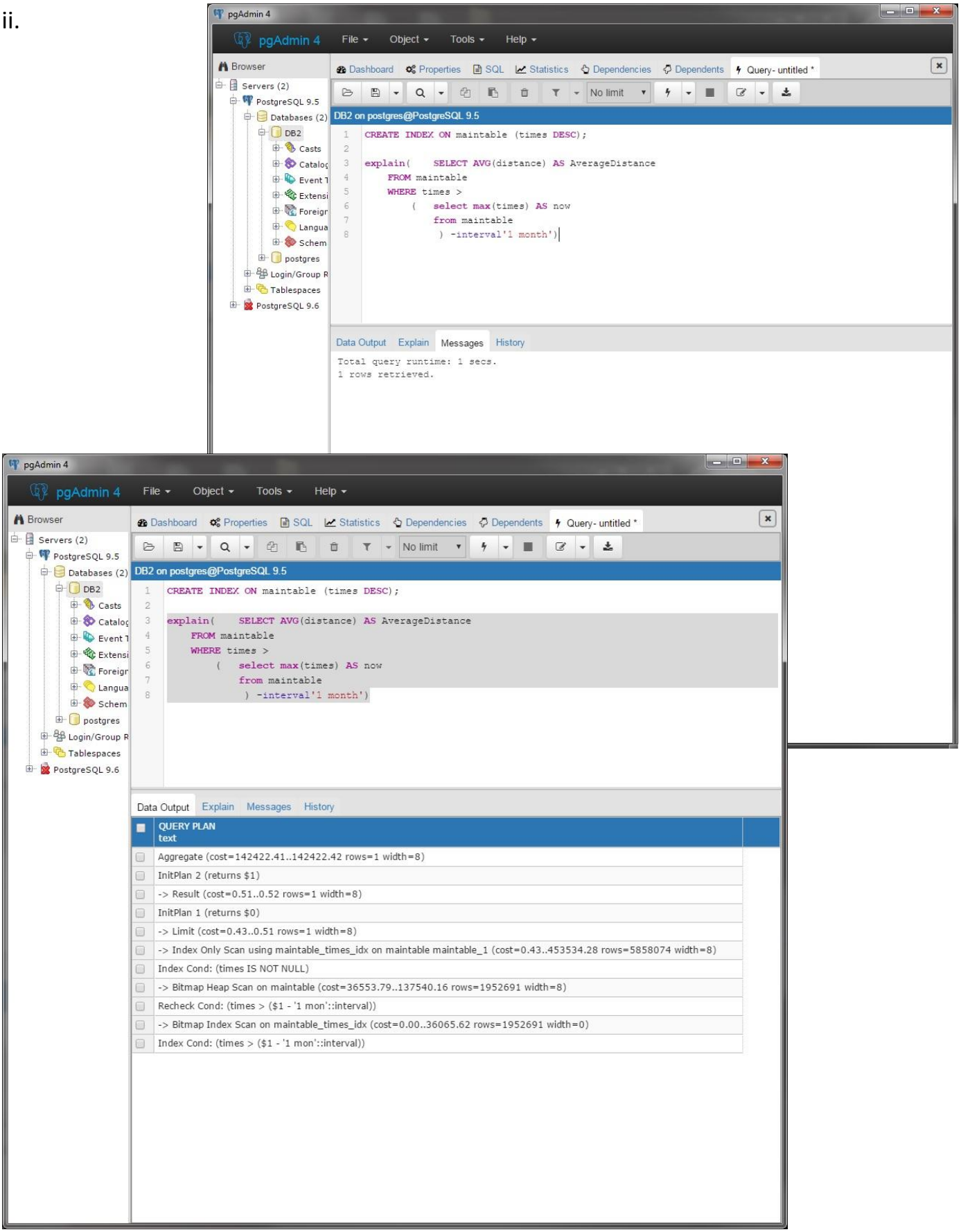
Σε αυτό το ερώτημα κλήθηκα να δημιουργήσω κάποια ευρετήρια όπου αυτά χρειαζόντουσαν. Έτσι δημιούργησα ένα B δέντρο στην στήλη με τις ημερομηνίες διότι μόνο τα ερωτήματα I,II μπορούν να επωφεληθούν από την χρήση δέντρων αφού τα άλλα 3 ερωτήματα αναγκαστικά θα αναγκαστούν να διαβάσουν ολόκληρο τον πίνακα αφού δεν υπάρχει κάποια συνθήκη σε αυτά τα SELECT. Διάλεξα B δέντρο αφού χρειάζεται να μελετήσουμε διάστημα στην βάση και τα B δέντρα μας δίνουν αυτή την δυνατότητα. Η εντολή που χρησιμοποίησα ήταν αυτή: **CREATE INDEX ON mainTable (times DESC);** Και τα αποτελέσματα τα παρακάτω:

i.





ii.


pgAdmin 4
File
Object
Tools
Help
Browser
Servers (2)
PostgreSQL 9.5
Databases (2)
DB2
postgres
Login/Group R
Tablespaces
PostgreSQL 9.6
Dashboard
Properties
SQL
Statistics
Dependencies
Dependents
Query- untitled *
No limit
DB2 on postgres@PostgreSQL 9.5
CREATE INDEX ON maintable (times DESC);
explain(SELECT AVG(distance) AS AverageDistance
FROM maintable
WHERE times >
(select max(times) AS now
from maintable
) -interval'1 month')
Data Output
Explain
Messages
History
Total query runtime: 1 secs.
1 rows retrieved.
QUERY PLAN
text
Aggregate (cost=142422.41..142422.42 rows=1 width=8)
InitPlan 2 (returns $1)
-> Result (cost=0.51..0.52 rows=1 width=8)
InitPlan 1 (returns $0)
-> Limit (cost=0.43..0.51 rows=1 width=8)
-> Index Only Scan using maintable_times_idx on maintable maintable_1 (cost=0.43..453534.28 rows=5858074 width=8)
Index Cond: (times IS NOT NULL)
-> Bitmap Heap Scan on maintable (cost=36553.79..137540.16 rows=1952691 width=8)
Recheck Cond: (times > ($1 - '1 mon'::interval))
-> Bitmap Index Scan on maintable_times_idx (cost=0.00..36065.62 rows=1952691 width=0)
Index Cond: (times > ($1 - '1 mon'::interval))

- Ε. Ερώτημα

Σε αυτό το ερώτημα έπρεπε να σπάσω το dataset σε κάποια partitions. Διάλεξα να το κάνω με range και να το χωρίσω ανά μήνα. Διάλεξα τον μήνα σαν κύρια στήλη για να κάνω αυτή την διαδικασία αφού και πάλι πιο πολύ θα επωφεληθούν τα δύο πρώτα queries από αυτό.

Αυτό που έκανα αρχικά είναι να φτιάξω ένα πίνακα ίδιο με αυτόν που ήδη είχα. Μετά έφτιαξα 8 υποπίνακες οι οποίοι κληρονομούσαν τον αρχικό αλλά ο καθένας είχε ένα συγκεκριμένο εύρος ημερομηνιών. Στην συνέχεια έφτιαξα ευρετήρια για κάθε ένα από τους υποπίνακες και ένα function και ένα trigger πριν το insert για να μπορέσω να βάζω τις σωστές εγγραφές στους αντίστοιχους πίνακες. Τέλος πέρασα τα δεδομένα στον πρώτο πίνακα που έφτιαξα και με την βοήθεια των trigger αυτά κατανεμήθηκαν στους αντίστοιχους υποπίνακες.

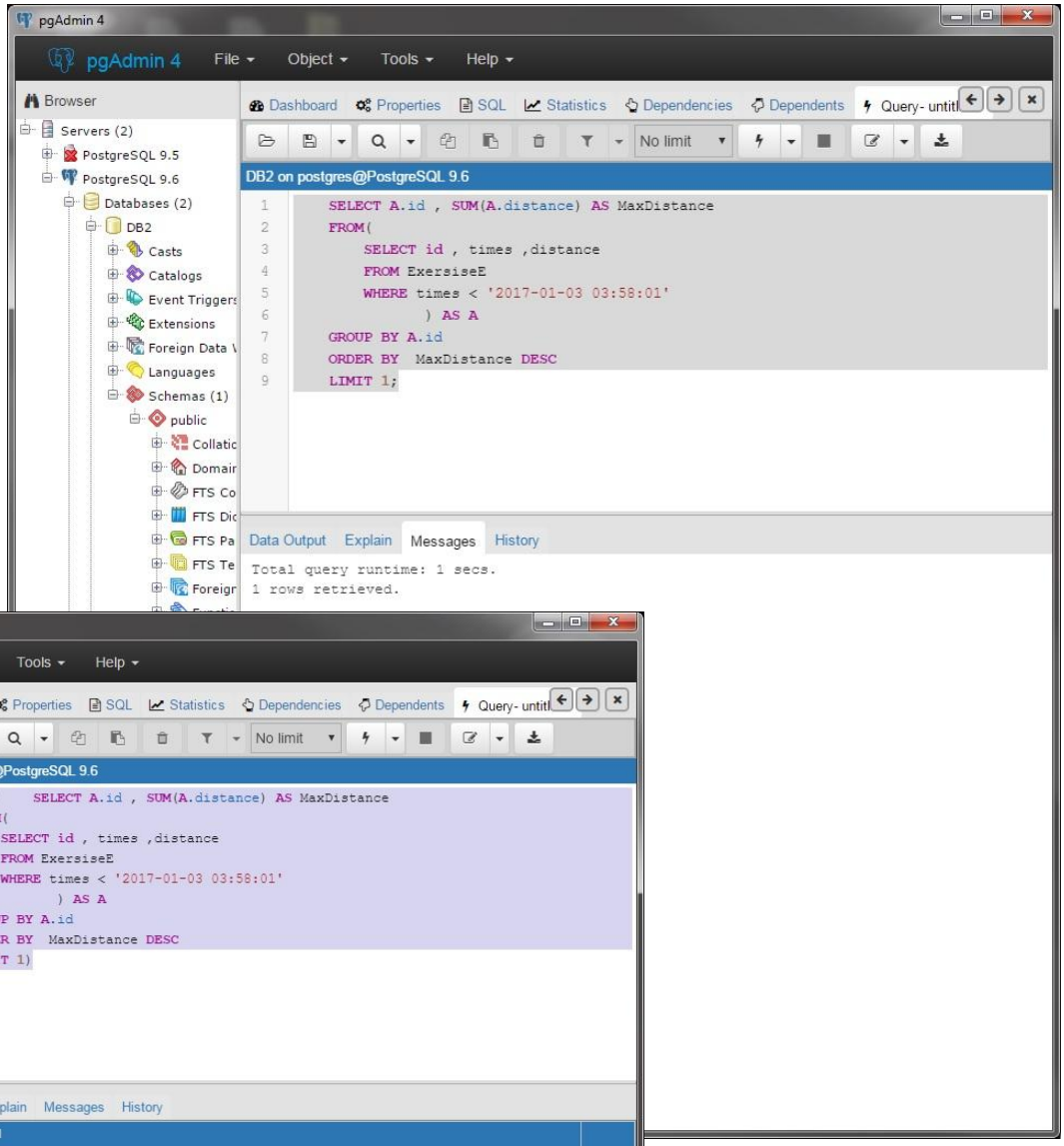
Ο κώδικας για όλα αυτά που προανέφερα είναι ο ακόλουθος:

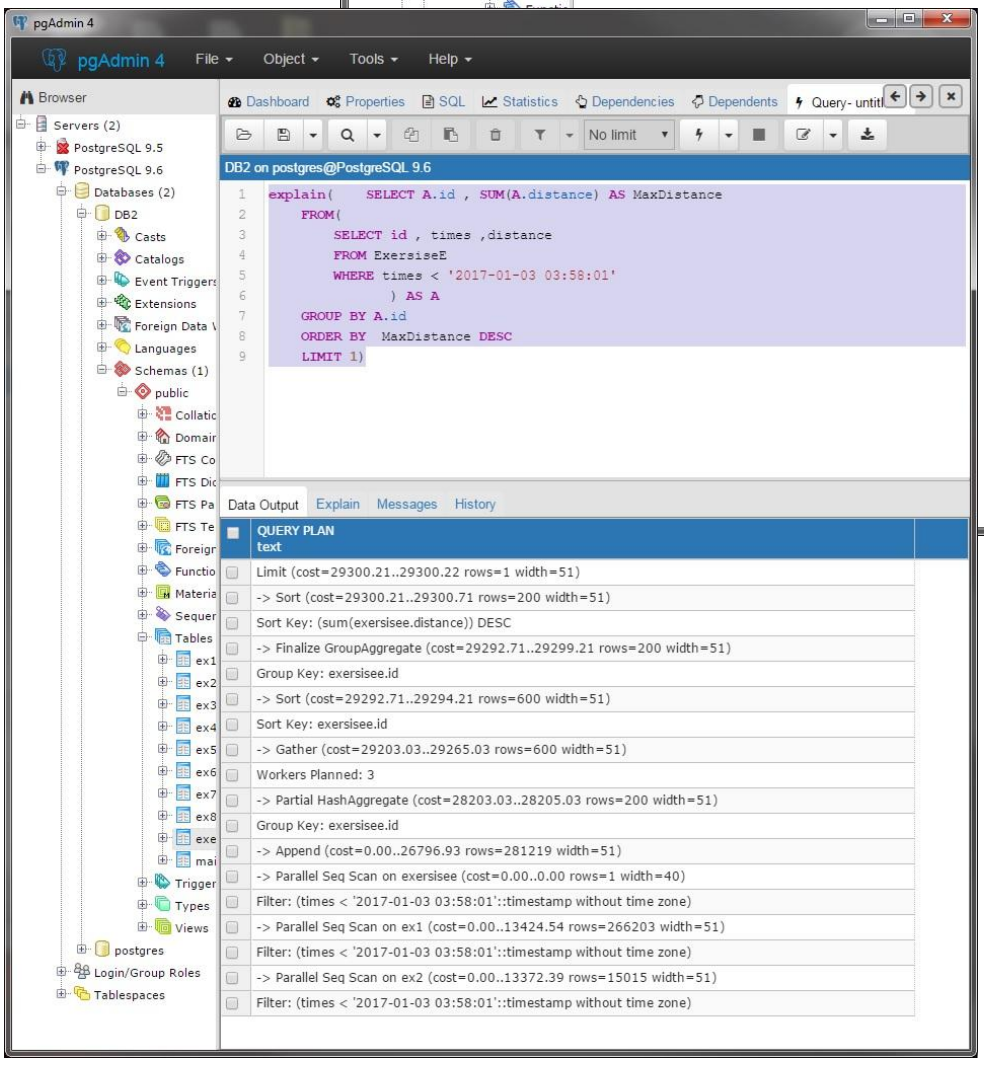
```
CREATE TABLE ExersiseE (
        id varchar,
        times timestamp,
        distance double precision,
        region_id smallint
);
CREATE TABLE Ex1 (
   CHECK ( times >= '2016-12-01 00:00:00' AND times < '2017-01-01 00:00:00' )
) INHERITS (ExersiseE);

CREATE TABLE Ex2 (
   CHECK ( times >= '2017-01-01 00:00:00' AND times < '2017-02-01 00:00:00' )
) INHERITS (ExersiseE);

CREATE TABLE Ex3 (
   CHECK ( times >= '2017-02-01 00:00:00' AND times < '2017-03-01 00:00:00' )
) INHERITS (ExersiseE);

CREATE TABLE Ex4 (
   CHECK ( times >= '2017-03-01 00:00:00' AND times < '2017-04-01 00:00:00' )
) INHERITS (ExersiseE);

CREATE TABLE Ex5 (
   CHECK ( times >= '2017-04-01 00:00:00' AND times < '2017-05-01 00:00:00' )
) INHERITS (ExersiseE);

CREATE TABLE Ex6 (
   CHECK ( times >= '2017-05-01 00:00:00' AND times < '2017-06-01 00:00:00' )
) INHERITS (ExersiseE);

CREATE TABLE Ex7 (
   CHECK ( times >= '2017-06-01 00:00:00' AND times < '2017-07-01 00:00:00' )
) INHERITS (ExersiseE);
```

```
CREATE TABLE Ex8 (
   CHECK (times >= '2017-07-01 00:00:00' AND times < '2017-08-01 00:00:00' )
) INHERITS (ExersiseE);

CREATE INDEX Ex1_index ON Ex1 (times);
CREATE INDEX Ex2_index ON Ex2 (times);
CREATE INDEX Ex3_index ON Ex3 (times);
CREATE INDEX Ex4_index ON Ex4 (times);
CREATE INDEX Ex5_index ON Ex5 (times);
CREATE INDEX Ex6_index ON Ex6 (times);
CREATE INDEX Ex7_index ON Ex7 (times);
CREATE INDEX Ex8_index ON Ex8 (times);

CREATE OR REPLACE FUNCTION timesInsertTrigger()
RETURNS TRIGGER AS $$
BEGIN
   IF (NEW.times >= '2016-12-01 00:00:00' AND NEW.times < '2017-01-01 00:00:00') THEN
      INSERT INTO Ex1 VALUES (NEW.*);
   ELSIF (NEW.times >= '2017-01-01 00:00:00' AND NEW.times < '2017-02-01 00:00:00') THEN
      INSERT INTO Ex2 VALUES (NEW.*);
   ELSIF (NEW.times >= '2017-02-01 00:00:00' AND NEW.times < '2017-03-01 00:00:00') THEN
      INSERT INTO Ex3 VALUES (NEW.*);
   ELSIF (NEW.times >= '2017-03-01 00:00:00' AND NEW.times < '2017-04-01 00:00:00') THEN
      INSERT INTO Ex4 VALUES (NEW.*);
    ELSIF (NEW.times >= '2017-04-01 00:00:00' AND NEW.times < '2017-05-01 00:00:00') THEN
      INSERT INTO Ex5 VALUES (NEW.*);
   ELSIF (NEW.times >= '2017-05-01 00:00:00' AND NEW.times < '2017-06-01 00:00:00') THEN
      INSERT INTO Ex6 VALUES (NEW.*);
   ELSIF (NEW.times >= '2017-06-01 00:00:00' AND NEW.times < '2017-07-01 00:00:00') THEN
      INSERT INTO Ex7 VALUES (NEW.*);
   ELSIF (NEW.times >= '2017-07-01 00:00:00' AND NEW.times < '2017-08-01 00:00:00') THEN
      INSERT INTO Ex8 VALUES (NEW.*);
  ELSE
      RAISE EXCEPTION 'Date out of range.';
   END IF;
   RETURN NULL;
END;
$$
LANGUAGE plpgsql;

CREATE TRIGGER inserTimesTrigger
   BEFORE INSERT ON ExersiseE
   FOR EACH ROW EXECUTE PROCEDURE  timesInsertTrigger();

COPY       ExersiseE(id, times, distance, region_id)
FROM       'C:\Users\George\Desktop\unipi\5o\db2'
WITH CSV HEADER;
```

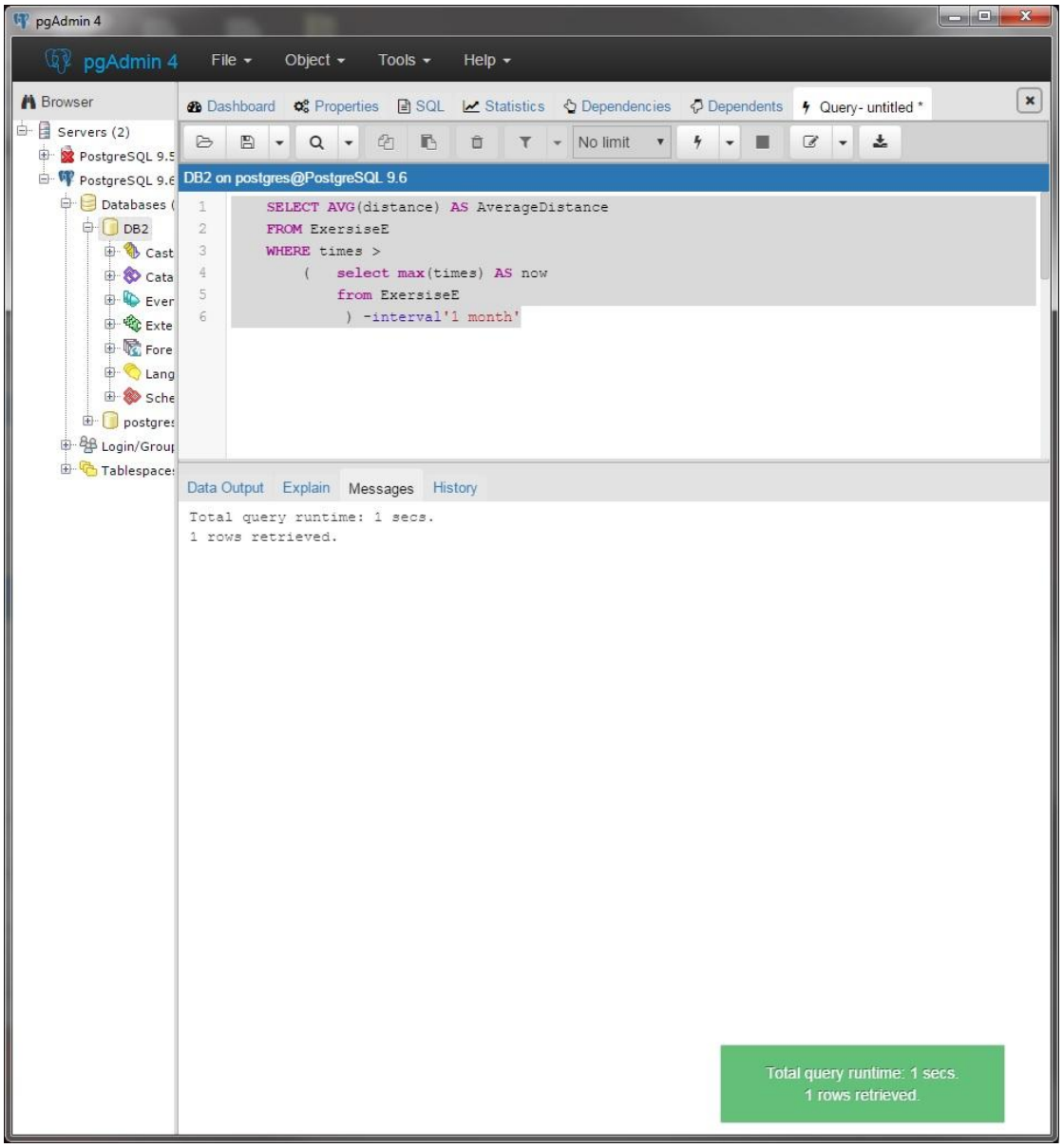
Τέλος, έτρεξα τα queries του ερωτήματος Α και είχα τα ακόλουθα αποτελέσματα(πάλι όπως και στα προηγούμενα ερωτήματα έκρινα σωστό να δείξω τους χρόνους και όχι το αποτέλεσμα καθεαυτό όπως και τα explain για κάθε query).

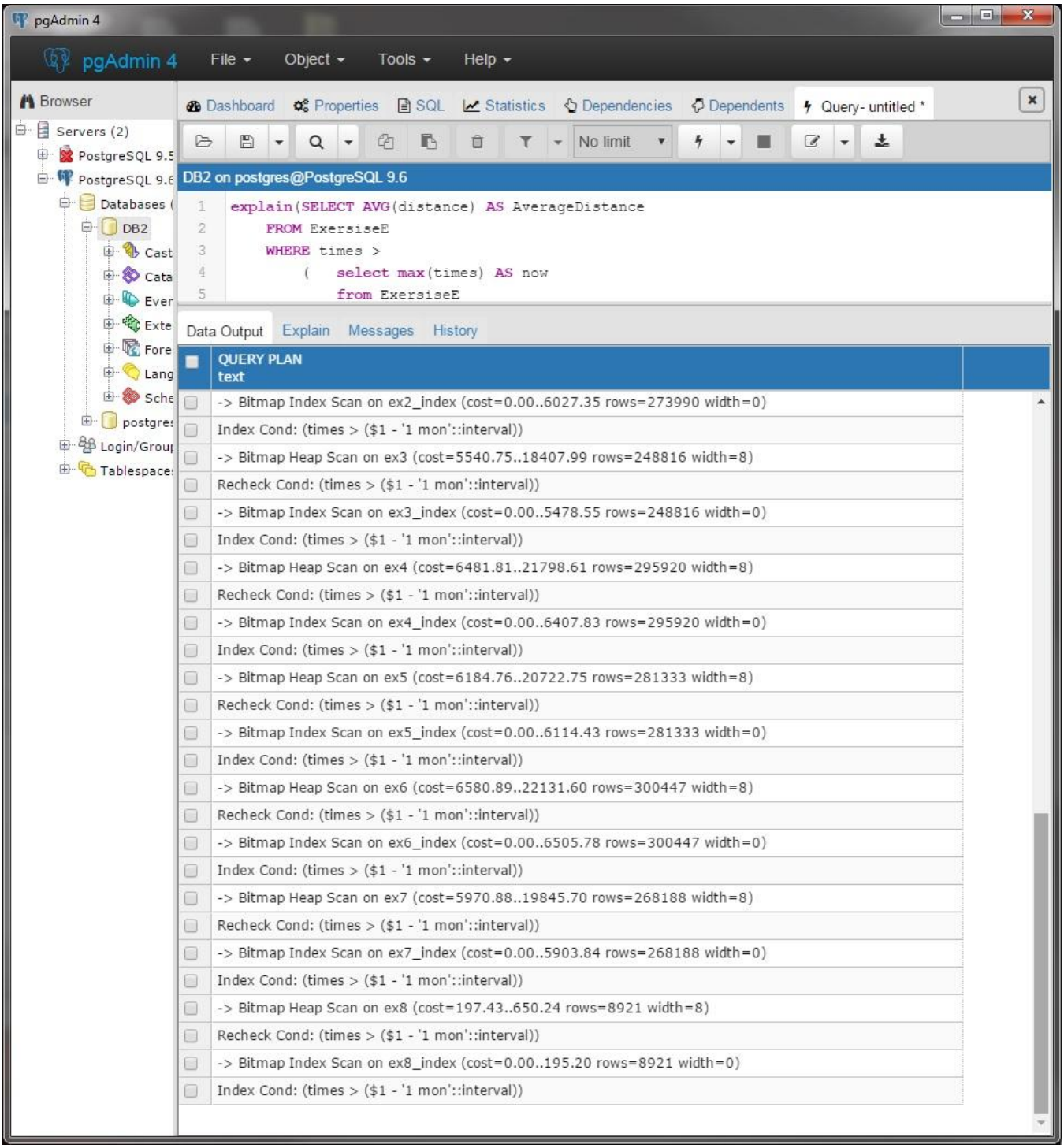
I.

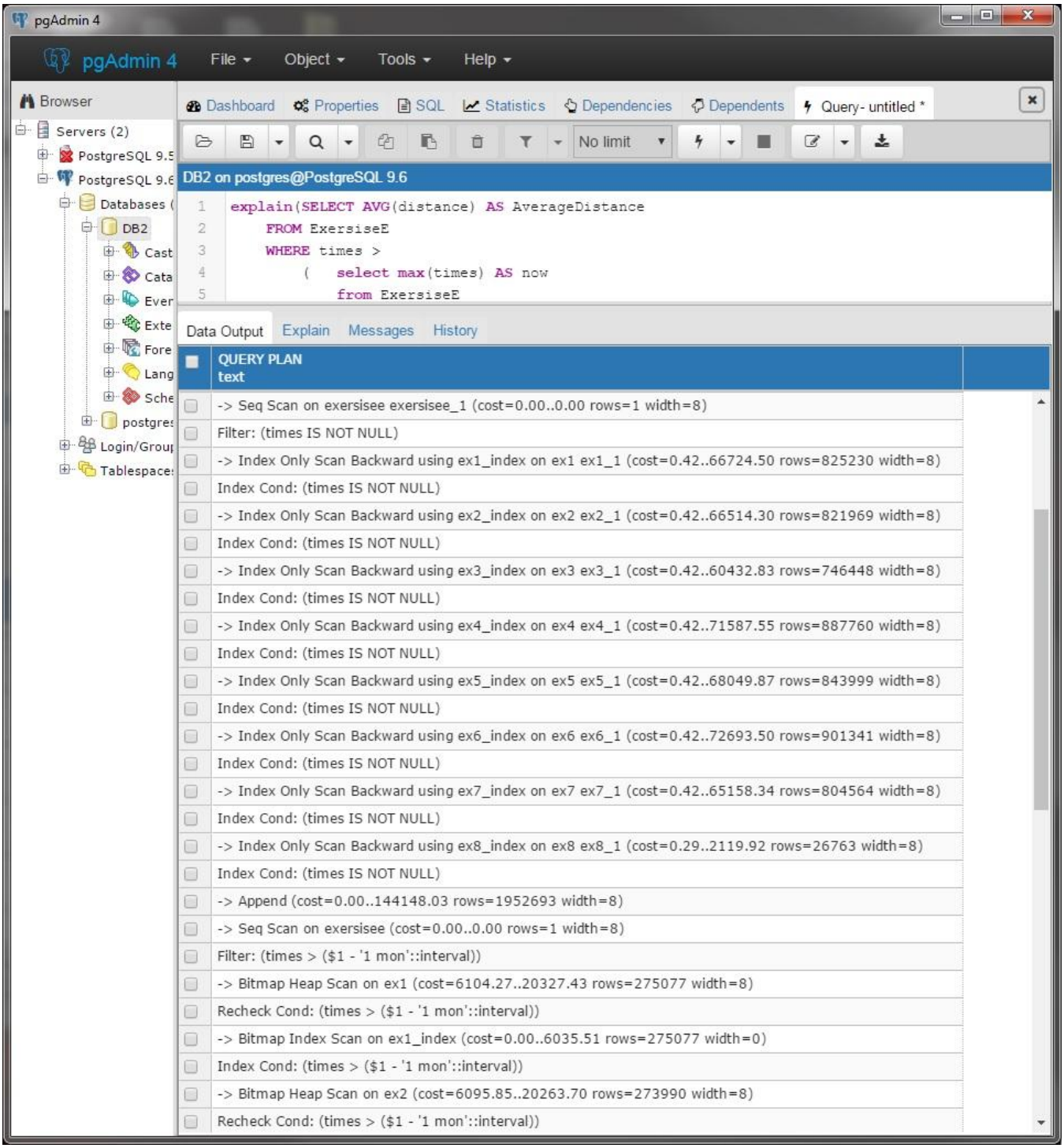


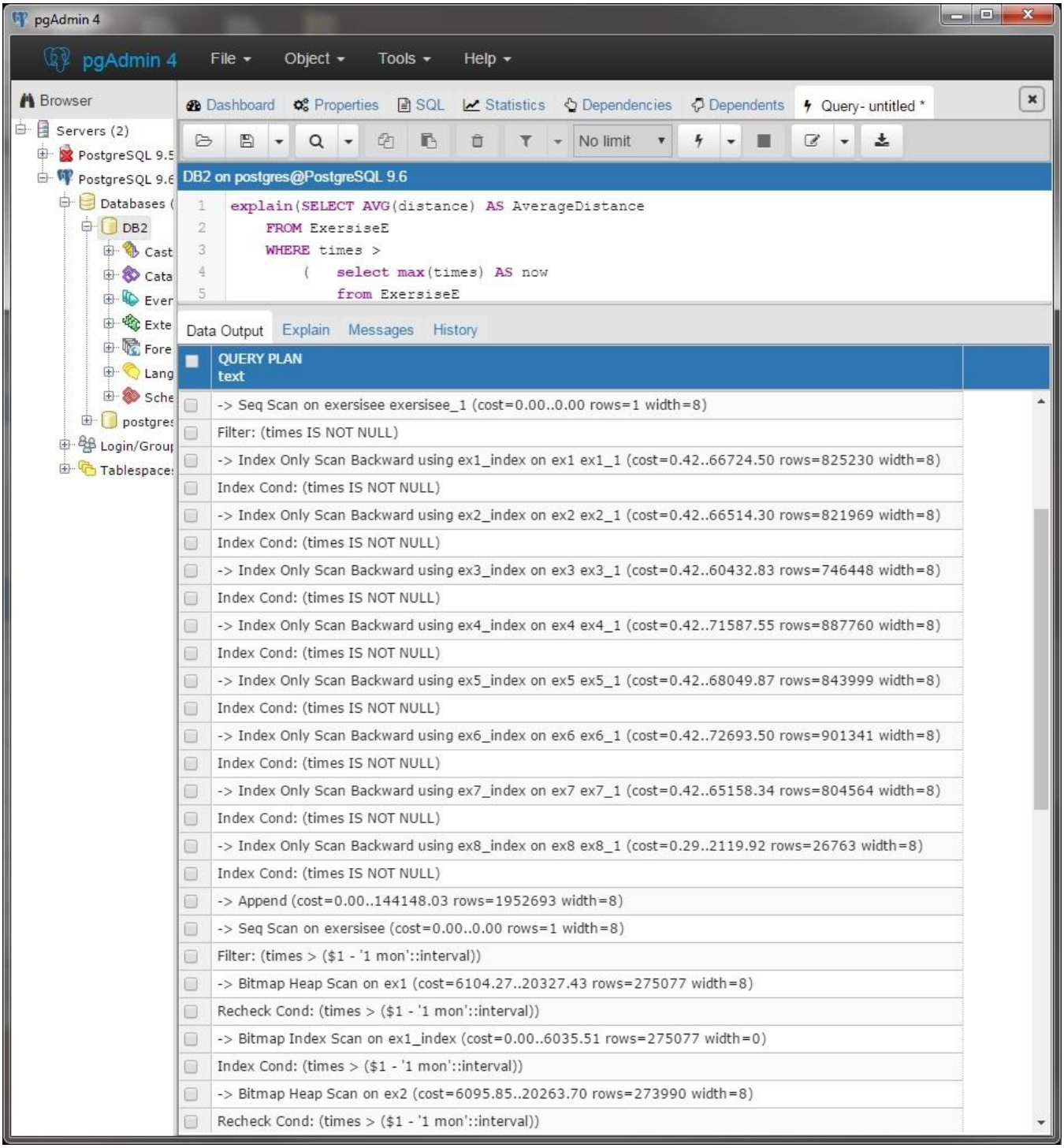


II. Το ερώτημα 2 λόγο της δομής του και του τρόπου που ορίσαμε τώρα τους πίνακες έχει ένα πολύ μεγάλο explain σε σχέση με τα υπόλοιπα, αυτό συμβαίνει διότι χρειάζεται να περάσει παραπάνω από μια φορά από την δομή που σχηματίσαμε παραπάνω. Το explain θα το δείξω σε τρία screenshot:

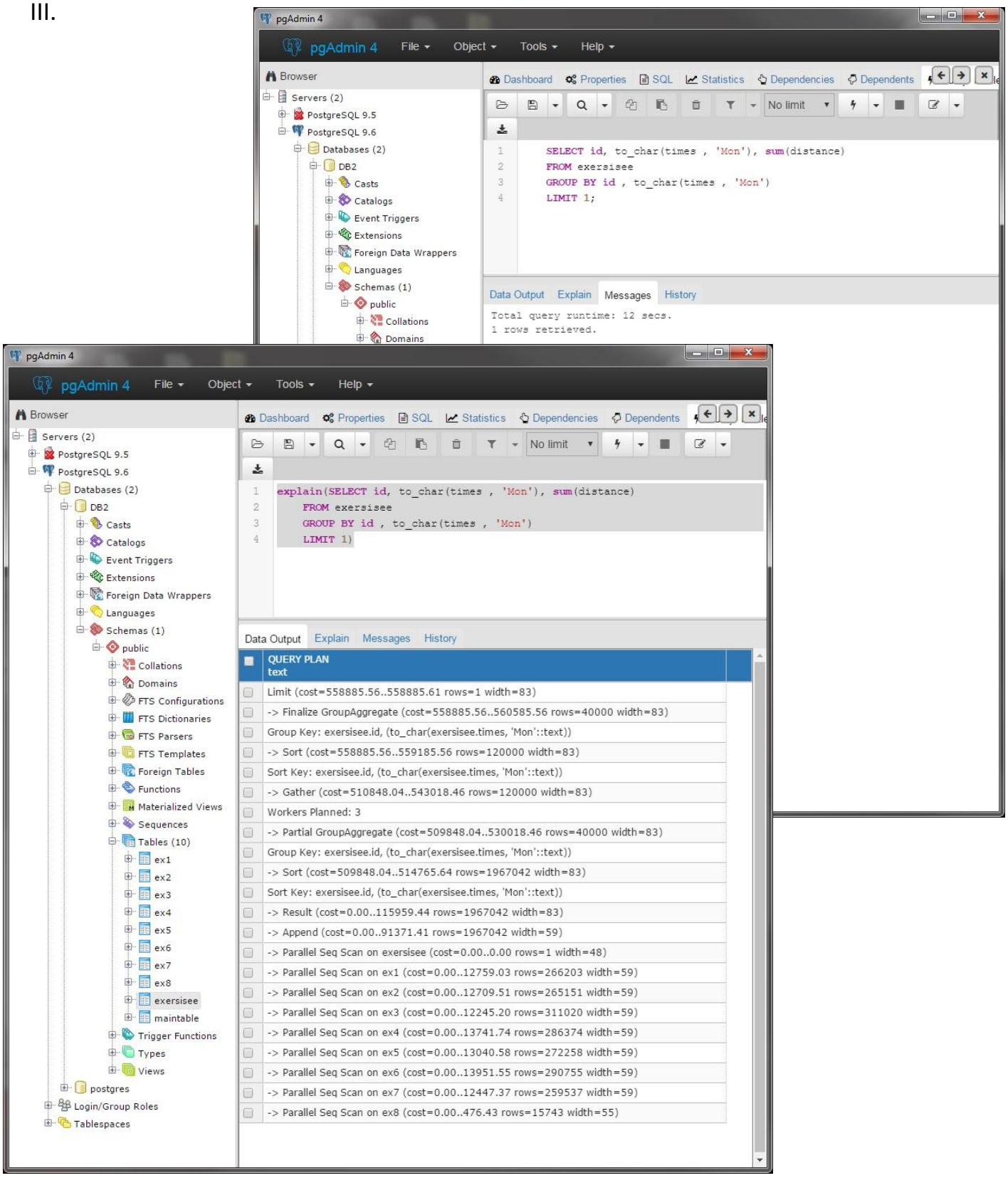



pgAdmin 4
File
Object
Tools
Help
Browser
Servers (2)
PostgreSQL 9.5
PostgreSQL 9.6
Databases (
DB2
Cast
Cata
Ever
Exte
Fore
Lang
Sche
postgres
Login/Group
Tablespaces
Dashboard
Properties
SQL
Statistics
Dependencies
Dependents
Query- untitled *
No limit
DB2 on postgres@PostgreSQL 9.6
1 explain(SELECT AVG(distance) AS AverageDistance
2 FROM ExersiseE
3 WHERE times >
4 (select max(times) AS now
5 from ExersiseE
Data Output
Explain
Messages
History
QUERY PLAN
text
-> Bitmap Index Scan on ex2_index (cost=0.00..6027.35 rows=273990 width=0)
Index Cond: (times > ($1 - '1 mon'::interval))
-> Bitmap Heap Scan on ex3 (cost=5540.75..18407.99 rows=248816 width=8)
Recheck Cond: (times > ($1 - '1 mon'::interval))
-> Bitmap Index Scan on ex3_index (cost=0.00..5478.55 rows=248816 width=0)
Index Cond: (times > ($1 - '1 mon'::interval))
-> Bitmap Heap Scan on ex4 (cost=6481.81..21798.61 rows=295920 width=8)
Recheck Cond: (times > ($1 - '1 mon'::interval))
-> Bitmap Index Scan on ex4_index (cost=0.00..6407.83 rows=295920 width=0)
Index Cond: (times > ($1 - '1 mon'::interval))
-> Bitmap Heap Scan on ex5 (cost=6184.76..20722.75 rows=281333 width=8)
Recheck Cond: (times > ($1 - '1 mon'::interval))
-> Bitmap Index Scan on ex5_index (cost=0.00..6114.43 rows=281333 width=0)
Index Cond: (times > ($1 - '1 mon'::interval))
-> Bitmap Heap Scan on ex6 (cost=6580.89..22131.60 rows=300447 width=8)
Recheck Cond: (times > ($1 - '1 mon'::interval))
-> Bitmap Index Scan on ex6_index (cost=0.00..6505.78 rows=300447 width=0)
Index Cond: (times > ($1 - '1 mon'::interval))
-> Bitmap Heap Scan on ex7 (cost=5970.88..19845.70 rows=268188 width=8)
Recheck Cond: (times > ($1 - '1 mon'::interval))
-> Bitmap Index Scan on ex7_index (cost=0.00..5903.84 rows=268188 width=0)
Index Cond: (times > ($1 - '1 mon'::interval))
-> Bitmap Heap Scan on ex8 (cost=197.43..650.24 rows=8921 width=8)
Recheck Cond: (times > ($1 - '1 mon'::interval))
-> Bitmap Index Scan on ex8_index (cost=0.00..195.20 rows=8921 width=0)
Index Cond: (times > ($1 - '1 mon'::interval))

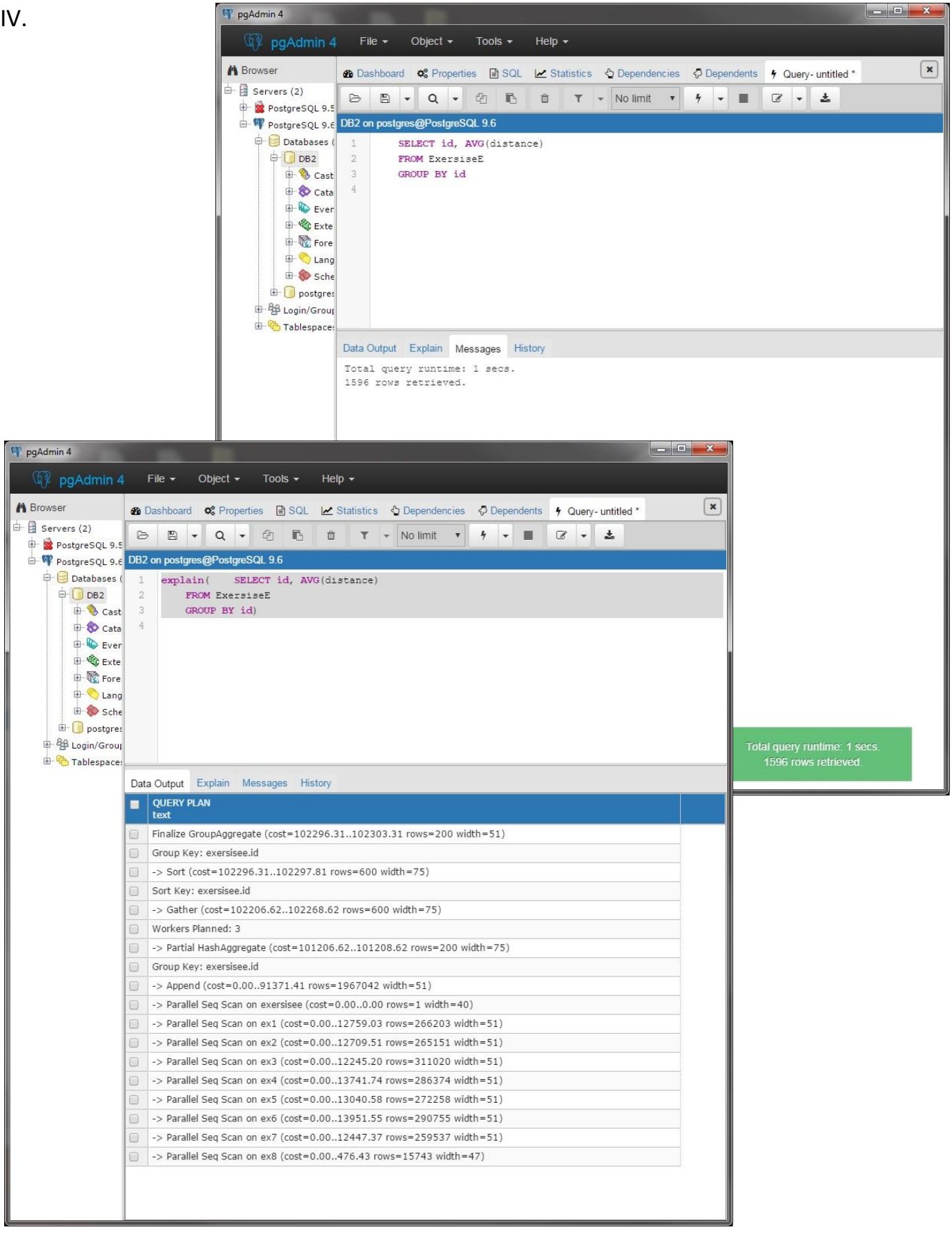

pgAdmin 4
File
Object
Tools
Help
Browser
Servers (2)
PostgreSQL 9.5
PostgreSQL 9.6
Databases (
DB2
Cast
Cata
Ever
Exte
Fore
Lang
Sche
postgres
Login/Group
Tablespaces
Dashboard
Properties
SQL
Statistics
Dependencies
Dependents
Query- untitled *
No limit
DB2 on postgres@PostgreSQL 9.6
1 explain(SELECT AVG(distance) AS AverageDistance
2 FROM ExersiseE
3 WHERE times >
4 (select max(times) AS now
5 from ExersiseE
Data Output
Explain
Messages
History
QUERY PLAN
text
-> Seq Scan on exersisee exersisee_1 (cost=0.00..0.00 rows=1 width=8)
Filter: (times IS NOT NULL)
-> Index Only Scan Backward using ex1_index on ex1 ex1_1 (cost=0.42..66724.50 rows=825230 width=8)
Index Cond: (times IS NOT NULL)
-> Index Only Scan Backward using ex2_index on ex2 ex2_1 (cost=0.42..66514.30 rows=821969 width=8)
Index Cond: (times IS NOT NULL)
-> Index Only Scan Backward using ex3_index on ex3 ex3_1 (cost=0.42..60432.83 rows=746448 width=8)
Index Cond: (times IS NOT NULL)
-> Index Only Scan Backward using ex4_index on ex4 ex4_1 (cost=0.42..71587.55 rows=887760 width=8)
Index Cond: (times IS NOT NULL)
-> Index Only Scan Backward using ex5_index on ex5 ex5_1 (cost=0.42..68049.87 rows=843999 width=8)
Index Cond: (times IS NOT NULL)
-> Index Only Scan Backward using ex6_index on ex6 ex6_1 (cost=0.42..72693.50 rows=901341 width=8)
Index Cond: (times IS NOT NULL)
-> Index Only Scan Backward using ex7_index on ex7 ex7_1 (cost=0.42..65158.34 rows=804564 width=8)
Index Cond: (times IS NOT NULL)
-> Index Only Scan Backward using ex8_index on ex8 ex8_1 (cost=0.29..2119.92 rows=26763 width=8)
Index Cond: (times IS NOT NULL)
-> Append (cost=0.00..144148.03 rows=1952693 width=8)
-> Seq Scan on exersisee (cost=0.00..0.00 rows=1 width=8)
Filter: (times > ($1 - '1 mon'::interval))
-> Bitmap Heap Scan on ex1 (cost=6104.27..20327.43 rows=275077 width=8)
Recheck Cond: (times > ($1 - '1 mon'::interval))
-> Bitmap Index Scan on ex1_index (cost=0.00..6035.51 rows=275077 width=0)
Index Cond: (times > ($1 - '1 mon'::interval))
-> Bitmap Heap Scan on ex2 (cost=6095.85..20263.70 rows=273990 width=8)
Recheck Cond: (times > ($1 - '1 mon'::interval))


pgAdmin 4
pgAdmin 4
File
Object
Tools
Help
Browser
Servers (2)
PostgreSQL 9.5
PostgreSQL 9.6
Databases (
DB2
Cast
Cata
Ever
Exte
Fore
Lang
Sche
postgres
Login/Group
Tablespaces
Dashboard
Properties
SQL
Statistics
Dependencies
Dependents
Query- untitled *
No limit
DB2 on postgres@PostgreSQL 9.6
1 explain(SELECT AVG(distance) AS AverageDistance
2 FROM ExersiseE
3 WHERE times >
4 (select max(times) AS now
5 from ExersiseE
Data Output
Explain
Messages
History
QUERY PLAN
text
-> Seq Scan on exersisee exersisee_1 (cost=0.00..0.00 rows=1 width=8)
Filter: (times IS NOT NULL)
-> Index Only Scan Backward using ex1_index on ex1 ex1_1 (cost=0.42..66724.50 rows=825230 width=8)
Index Cond: (times IS NOT NULL)
-> Index Only Scan Backward using ex2_index on ex2 ex2_1 (cost=0.42..66514.30 rows=821969 width=8)
Index Cond: (times IS NOT NULL)
-> Index Only Scan Backward using ex3_index on ex3 ex3_1 (cost=0.42..60432.83 rows=746448 width=8)
Index Cond: (times IS NOT NULL)
-> Index Only Scan Backward using ex4_index on ex4 ex4_1 (cost=0.42..71587.55 rows=887760 width=8)
Index Cond: (times IS NOT NULL)
-> Index Only Scan Backward using ex5_index on ex5 ex5_1 (cost=0.42..68049.87 rows=843999 width=8)
Index Cond: (times IS NOT NULL)
-> Index Only Scan Backward using ex6_index on ex6 ex6_1 (cost=0.42..72693.50 rows=901341 width=8)
Index Cond: (times IS NOT NULL)
-> Index Only Scan Backward using ex7_index on ex7 ex7_1 (cost=0.42..65158.34 rows=804564 width=8)
Index Cond: (times IS NOT NULL)
-> Index Only Scan Backward using ex8_index on ex8 ex8_1 (cost=0.29..2119.92 rows=26763 width=8)
Index Cond: (times IS NOT NULL)
-> Append (cost=0.00..144148.03 rows=1952693 width=8)
-> Seq Scan on exersisee (cost=0.00..0.00 rows=1 width=8)
Filter: (times > ($1 - '1 mon'::interval))
-> Bitmap Heap Scan on ex1 (cost=6104.27..20327.43 rows=275077 width=8)
Recheck Cond: (times > ($1 - '1 mon'::interval))
-> Bitmap Index Scan on ex1_index (cost=0.00..6035.51 rows=275077 width=0)
Index Cond: (times > ($1 - '1 mon'::interval))
-> Bitmap Heap Scan on ex2 (cost=6095.85..20263.70 rows=273990 width=8)
Recheck Cond: (times > ($1 - '1 mon'::interval))

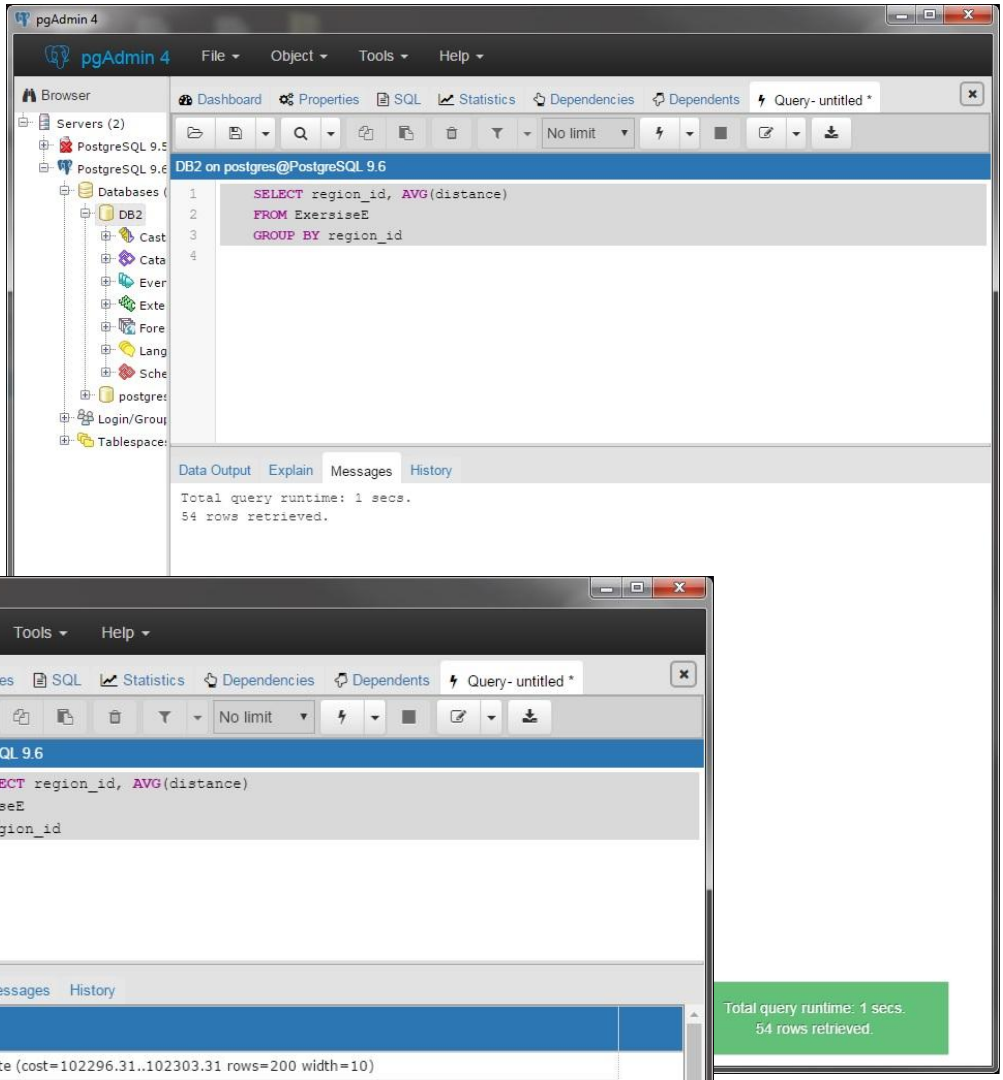
III.

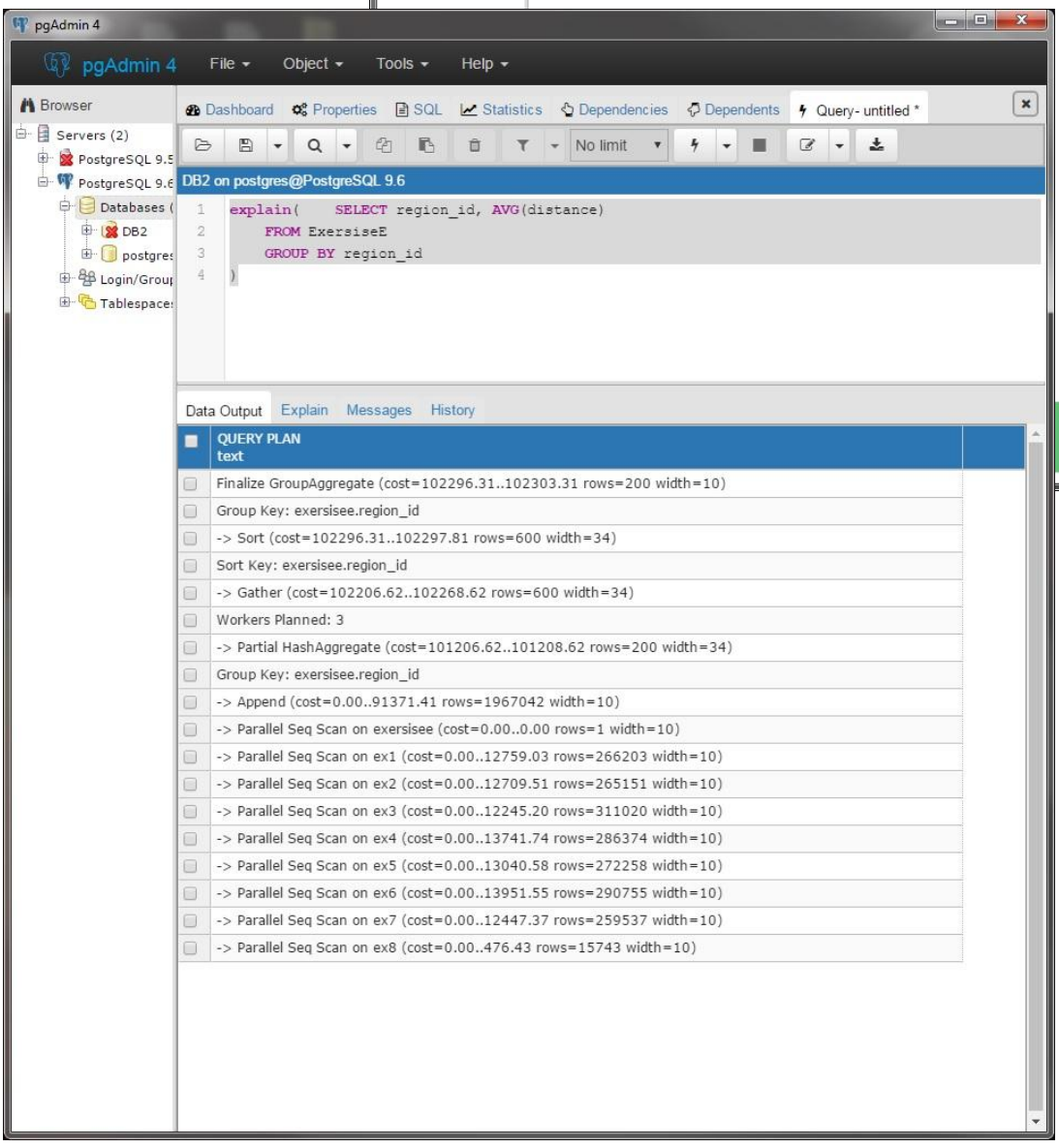

pgAdmin 4
File
Object
Tools
Help
Browser
Servers (2)
PostgreSQL 9.5
PostgreSQL 9.6
Databases (2)
DB2
Casts
Catalogs
Event Triggers
Extensions
Foreign Data Wrappers
Languages
Schemas (1)
public
Collations
Domains
FTS Configurations
FTS Dictionaries
FTS Parsers
FTS Templates
Foreign Tables
Functions
Materialized Views
Sequences
Tables (10)
ex1
ex2
ex3
ex4
ex5
ex6
ex7
ex8
exersisee
maintable
Trigger Functions
Types
Views
postgres
Login/Group Roles
Tablespaces
Dashboard
Properties
SQL
Statistics
Dependencies
Dependents
No limit
SELECT id, to_char(times , 'Mon'), sum(distance)
FROM exersisee
GROUP BY id , to_char(times , 'Mon')
LIMIT 1;
Data Output
Explain
Messages
History
Total query runtime: 12 secs.
1 rows retrieved.
explain(SELECT id, to_char(times , 'Mon'), sum(distance)
FROM exersisee
GROUP BY id , to_char(times , 'Mon')
LIMIT 1)
QUERY PLAN
text
Limit (cost=558885.56..558885.61 rows=1 width=83)
-> Finalize GroupAggregate (cost=558885.56..560585.56 rows=40000 width=83)
Group Key: exersisee.id, (to_char(exersisee.times, 'Mon'::text))
-> Sort (cost=558885.56..559185.56 rows=120000 width=83)
Sort Key: exersisee.id, (to_char(exersisee.times, 'Mon'::text))
-> Gather (cost=510848.04..543018.46 rows=120000 width=83)
Workers Planned: 3
-> Partial GroupAggregate (cost=509848.04..530018.46 rows=40000 width=83)
Group Key: exersisee.id, (to_char(exersisee.times, 'Mon'::text))
-> Sort (cost=509848.04..514765.64 rows=1967042 width=83)
Sort Key: exersisee.id, (to_char(exersisee.times, 'Mon'::text))
-> Result (cost=0.00..115959.44 rows=1967042 width=83)
-> Append (cost=0.00..91371.41 rows=1967042 width=59)
-> Parallel Seq Scan on exersisee (cost=0.00..0.00 rows=1 width=48)
-> Parallel Seq Scan on ex1 (cost=0.00..12759.03 rows=266203 width=59)
-> Parallel Seq Scan on ex2 (cost=0.00..12709.51 rows=265151 width=59)
-> Parallel Seq Scan on ex3 (cost=0.00..12245.20 rows=311020 width=59)
-> Parallel Seq Scan on ex4 (cost=0.00..13741.74 rows=286374 width=59)
-> Parallel Seq Scan on ex5 (cost=0.00..13040.58 rows=272258 width=59)
-> Parallel Seq Scan on ex6 (cost=0.00..13951.55 rows=290755 width=59)
-> Parallel Seq Scan on ex7 (cost=0.00..12447.37 rows=259537 width=59)
-> Parallel Seq Scan on ex8 (cost=0.00..476.43 rows=15743 width=55)

IV.


pgAdmin 4
File
Object
Tools
Help
Browser
Servers (2)
PostgreSQL 9.5
PostgreSQL 9.6
Databases (
DB2
Dashboard
Properties
SQL
Statistics
Dependencies
Dependents
Query- untitled *
No limit
DB2 on postgres@PostgreSQL 9.6
SELECT id, AVG(distance)
FROM ExersiseE
GROUP BY id
Data Output
Explain
Messages
History
Total query runtime: 1 secs.
1596 rows retrieved.
explain(SELECT id, AVG(distance)
FROM ExersiseE
GROUP BY id)
Total query runtime: 1 secs.
1596 rows retrieved.
QUERY PLAN
text
Finalize GroupAggregate (cost=102296.31..102303.31 rows=200 width=51)
Group Key: exersisee.id
-> Sort (cost=102296.31..102297.81 rows=600 width=75)
Sort Key: exersisee.id
-> Gather (cost=102206.62..102268.62 rows=600 width=75)
Workers Planned: 3
-> Partial HashAggregate (cost=101206.62..101208.62 rows=200 width=75)
Group Key: exersisee.id
-> Append (cost=0.00..91371.41 rows=1967042 width=51)
-> Parallel Seq Scan on exersisee (cost=0.00..0.00 rows=1 width=40)
-> Parallel Seq Scan on ex1 (cost=0.00..12759.03 rows=266203 width=51)
-> Parallel Seq Scan on ex2 (cost=0.00..12709.51 rows=265151 width=51)
-> Parallel Seq Scan on ex3 (cost=0.00..12245.20 rows=311020 width=51)
-> Parallel Seq Scan on ex4 (cost=0.00..13741.74 rows=286374 width=51)
-> Parallel Seq Scan on ex5 (cost=0.00..13040.58 rows=272258 width=51)
-> Parallel Seq Scan on ex6 (cost=0.00..13951.55 rows=290755 width=51)
-> Parallel Seq Scan on ex7 (cost=0.00..12447.37 rows=259537 width=51)
-> Parallel Seq Scan on ex8 (cost=0.00..476.43 rows=15743 width=47)

V.


pgAdmin 4
File
Object
Tools
Help
Browser
Dashboard
Properties
SQL
Statistics
Dependencies
Dependents
Query- untitled *
No limit
DB2 on postgres@PostgreSQL 9.6
SELECT region_id, AVG(distance)
FROM ExersiseE
GROUP BY region_id
Data Output
Explain
Messages
History
Total query runtime: 1 secs.
54 rows retrieved.

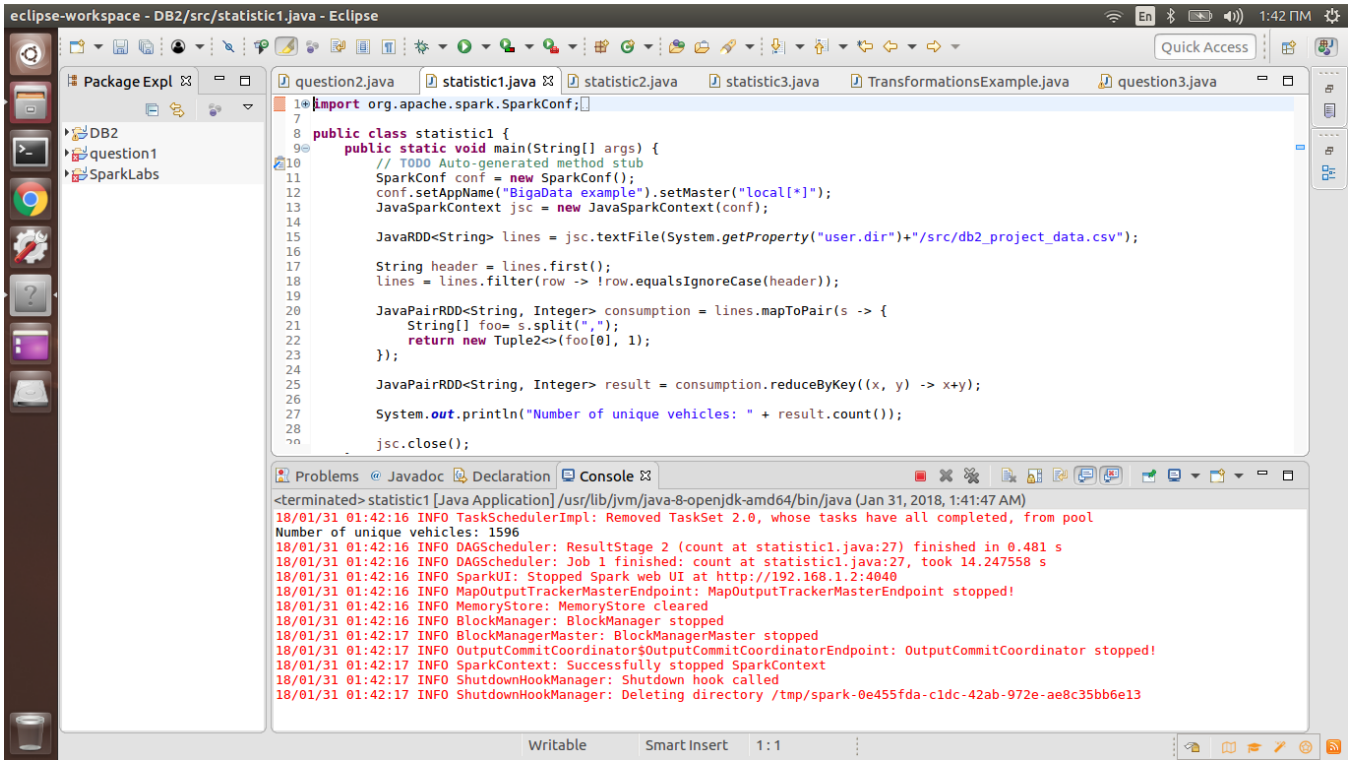

pgAdmin 4
File
Object
Tools
Help
Browser
Dashboard
Properties
SQL
Statistics
Dependencies
Dependents
Query- untitled *
No limit
DB2 on postgres@PostgreSQL 9.6
explain(SELECT region_id, AVG(distance)
FROM ExersiseE
GROUP BY region_id
)
Data Output
Explain
Messages
History
QUERY PLAN
text
Finalize GroupAggregate (cost=102296.31..102303.31 rows=200 width=10)
Group Key: exersisee.region_id
-> Sort (cost=102296.31..102297.81 rows=600 width=34)
Sort Key: exersisee.region_id
-> Gather (cost=102206.62..102268.62 rows=600 width=34)
Workers Planned: 3
-> Partial HashAggregate (cost=101206.62..101208.62 rows=200 width=34)
Group Key: exersisee.region_id
-> Append (cost=0.00..91371.41 rows=1967042 width=10)
-> Parallel Seq Scan on exersisee (cost=0.00..0.00 rows=1 width=10)
-> Parallel Seq Scan on ex1 (cost=0.00..12759.03 rows=266203 width=10)
-> Parallel Seq Scan on ex2 (cost=0.00..12709.51 rows=265151 width=10)
-> Parallel Seq Scan on ex3 (cost=0.00..12245.20 rows=311020 width=10)
-> Parallel Seq Scan on ex4 (cost=0.00..13741.74 rows=286374 width=10)
-> Parallel Seq Scan on ex5 (cost=0.00..13040.58 rows=272258 width=10)
-> Parallel Seq Scan on ex6 (cost=0.00..13951.55 rows=290755 width=10)
-> Parallel Seq Scan on ex7 (cost=0.00..12447.37 rows=259537 width=10)
-> Parallel Seq Scan on ex8 (cost=0.00..476.43 rows=15743 width=10)
Total query runtime: 1 secs.
54 rows retrieved.

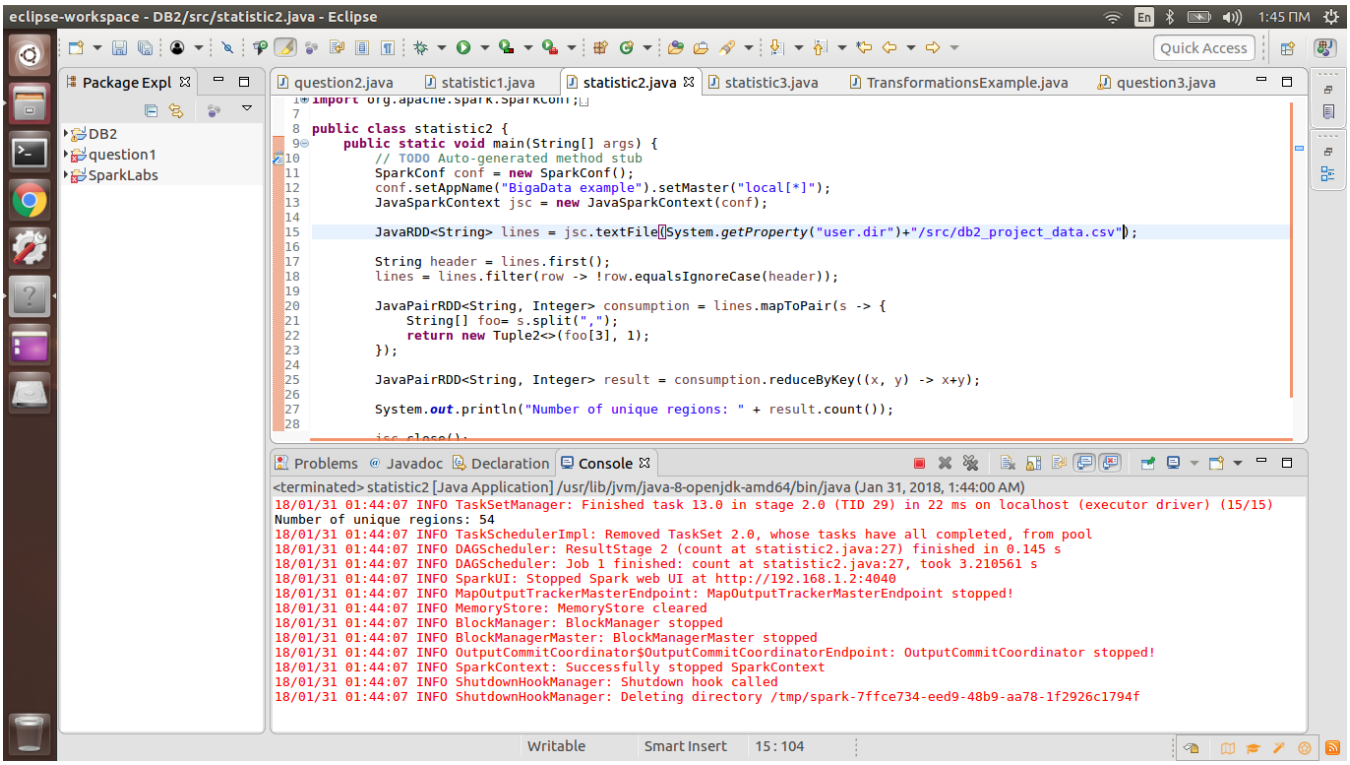
Μέρος 2ο

Α. Στο πρώτο ερώτημα δημιούργησα μερικά στατιστικά για την βάση, συγκεκριμένα το πρώτο μετρά το πλήθος των μοναδικών αυτοκινήτων, το δεύτερο μετρά το πλήθος των μοναδικών περιοχών και το τρίτο μετρά την συνολική απόσταση ανά περιοχή. Για το κάθε ένα έπραξα παρόμοια οπότε θα περιγράψω την λειτουργεί μόνο του πρώτου:

Αρχικά δημιούργησα ένα configuration, έθεσα ένα master και επίσης δημιούργησα το sparkcontext. Στην συνέχεια πέρασα το αρχείο με την βοήθεια του context στο JavaRDD και του αγνόησα την πρώτη γραμμή δηλαδή το header. Μετά έφτιαξα ένα PairRDD για να μπορέσω να κάνω το mapping. Για το mapping χρησιμοποίησα την .mapToPair και δημιούργησα ένα Tuple2 το οποίο γέμισα με την πρώτη και την τρίτη εγγραφή κάθε γραμμής αντίστοιχα. Τέλος για το reduceByKey έλαβα σαν key το id που είχα περάσει σαν πρώτο στοιχείο προηγουμένως και έκανα reduce με αθροιστική συνθήκη, ουσιαστικά η σύνθεση δεν έπαιζε πολύ ρόλο αφού στο τέλος θα έκανα count τις συνολικές γραμμές για να μου πει πόσοι είναι οι χρήστες.

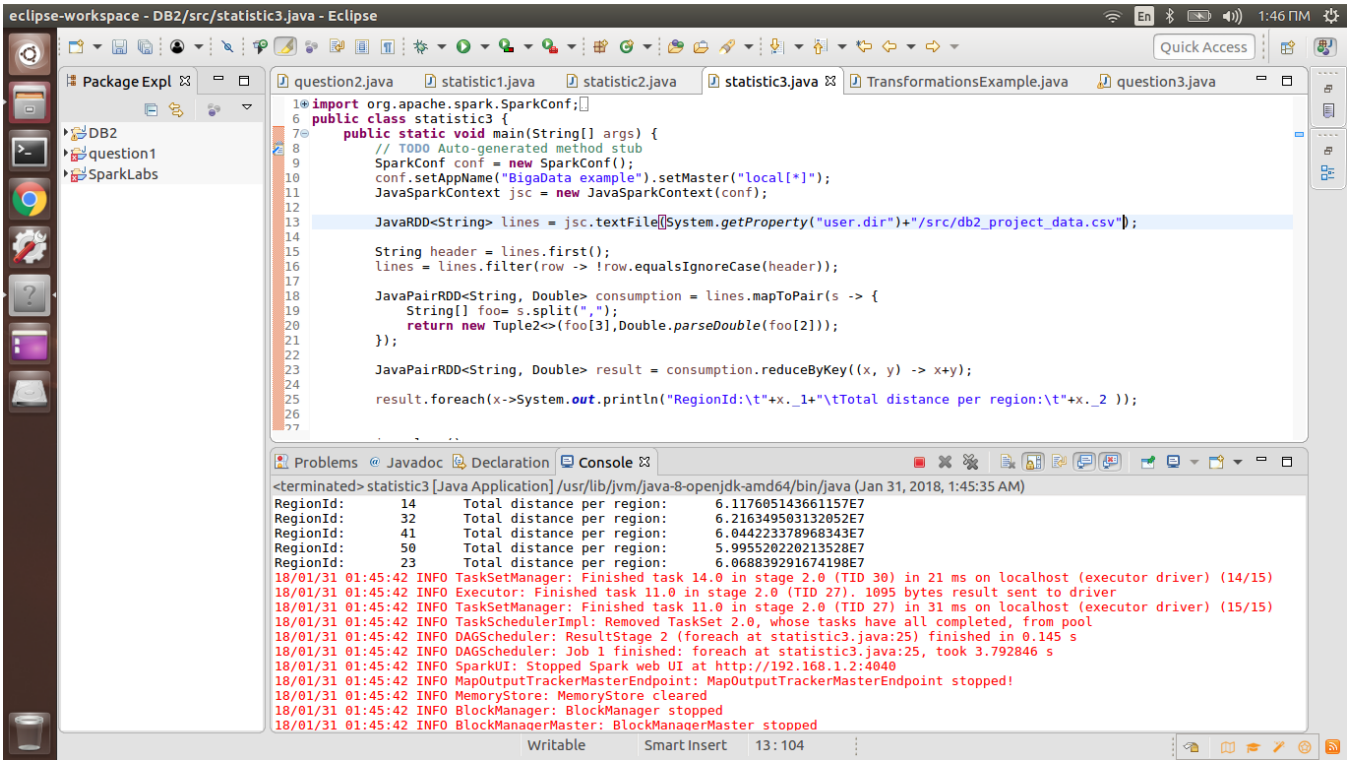
Παραθέτω Screenshot στην συνέχεια:






```
import org.apache.spark.SparkConf;

public class statistic2 {
    public static void main(String[] args) {
        // TODO Auto-generated method stub
        SparkConf conf = new SparkConf();
        conf.setAppName("BigaData example").setMaster("local[*]");
        JavaSparkContext jsc = new JavaSparkContext(conf);

        JavaRDD<String> lines = jsc.textFile(System.getProperty("user.dir")+"/src/db2_project_data.csv");

        String header = lines.first();
        lines = lines.filter(row -> !row.equalsIgnoreCase(header));

        JavaPairRDD<String, Integer> consumption = lines.mapToPair(s -> {
            String[] foo= s.split(",");
            return new Tuple2<>(foo[3], 1);
        });

        JavaPairRDD<String, Integer> result = consumption.reduceByKey((x, y) -> x+y);

        System.out.println("Number of unique regions: " + result.count());
```

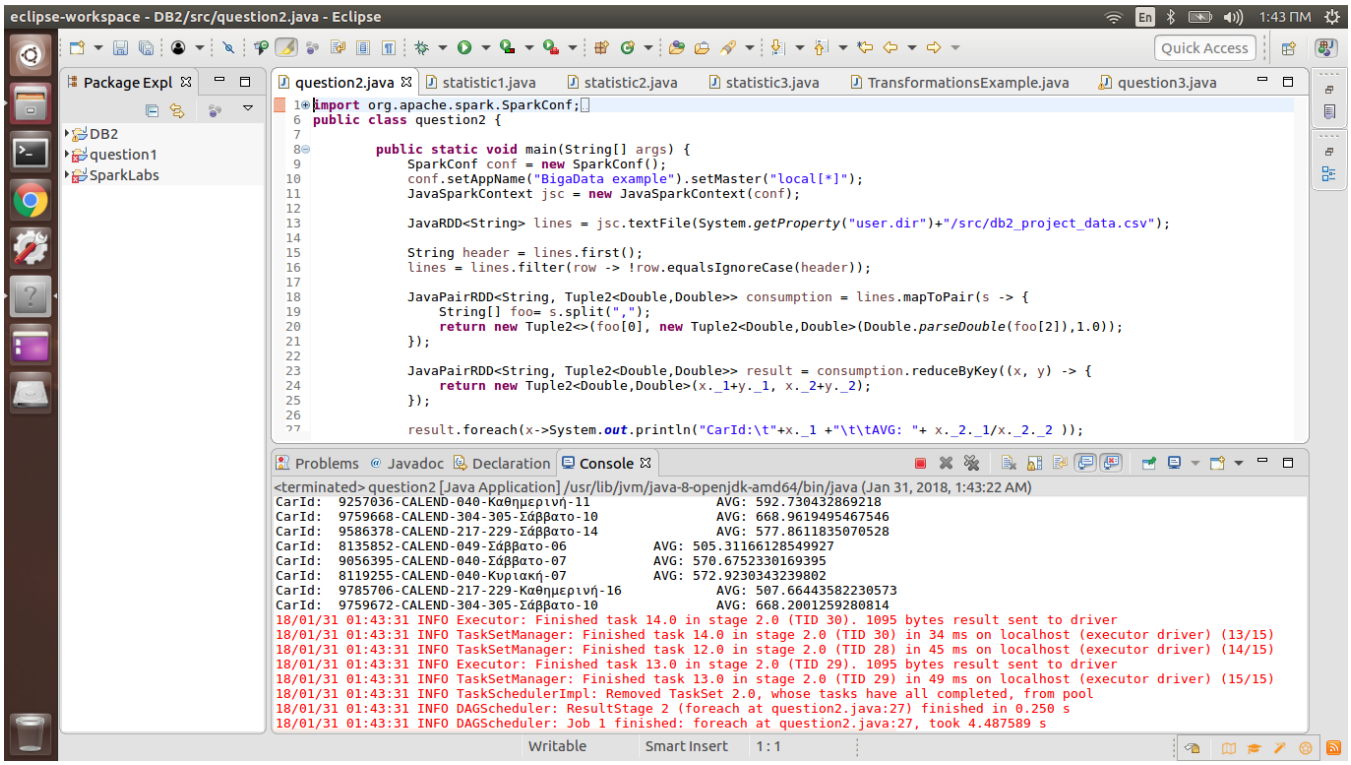
```
18/01/31 01:44:07 INFO TaskSetManager: Finished task 13.0 in stage 2.0 (TID 29) in 22 ms on localhost (executor driver) (15/15)
18/01/31 01:44:07 INFO TaskSchedulerImpl: Removed TaskSet 2.0, whose tasks have all completed, from pool
18/01/31 01:44:07 INFO DAGScheduler: ResultStage 2 (count at statistic2.java:27) finished in 0.145 s
18/01/31 01:44:07 INFO DAGScheduler: Job 1 finished: count at statistic2.java:27, took 3.210561 s
18/01/31 01:44:07 INFO SparkUI: Stopped Spark web UI at http://192.168.1.2:4040
18/01/31 01:44:07 INFO MapOutputTrackerMasterEndpoint: MapOutputTrackerMasterEndpoint stopped!
18/01/31 01:44:07 INFO MemoryStore: MemoryStore cleared
18/01/31 01:44:07 INFO BlockManager: BlockManager stopped
18/01/31 01:44:07 INFO BlockManagerMaster: BlockManagerMaster stopped
18/01/31 01:44:07 INFO OutputCommitCoordinator$OutputCommitCoordinatorEndpoint: OutputCommitCoordinator stopped!
18/01/31 01:44:07 INFO SparkContext: Successfully stopped SparkContext
18/01/31 01:44:07 INFO ShutdownHookManager: Shutdown hook called
18/01/31 01:44:07 INFO ShutdownHookManager: Deleting directory /tmp/spark-7ffce734-eed9-48b9-aa78-1f2926c1794f
```



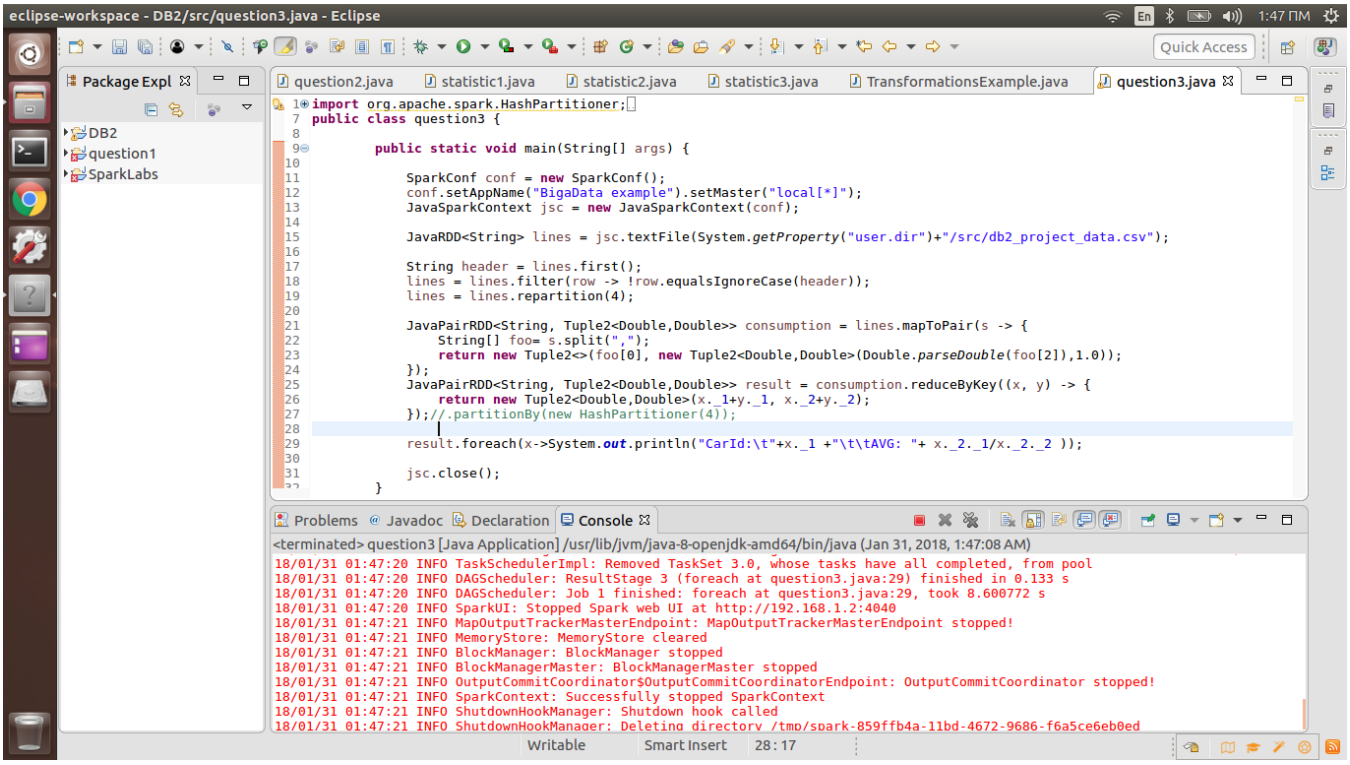

```
import org.apache.spark.SparkConf;
public class statistic3 {
    public static void main(String[] args) {
        // TODO Auto-generated method stub
        SparkConf conf = new SparkConf();
        conf.setAppName("BigaData example").setMaster("local[*]");
        JavaSparkContext jsc = new JavaSparkContext(conf);

        JavaRDD<String> lines = jsc.textFile(System.getProperty("user.dir")+"/src/db2_project_data.csv");

        String header = lines.first();
        lines = lines.filter(row -> !row.equalsIgnoreCase(header));

        JavaPairRDD<String, Double> consumption = lines.mapToPair(s -> {
            String[] foo= s.split(",");
            return new Tuple2<>(foo[3],Double.parseDouble(foo[2]));
        });

        JavaPairRDD<String, Double> result = consumption.reduceByKey((x, y) -> x+y);

        result.foreach(x->System.out.println("RegionId:\t"+x._1+"\tTotal distance per region:\t"+x._2 ));
```

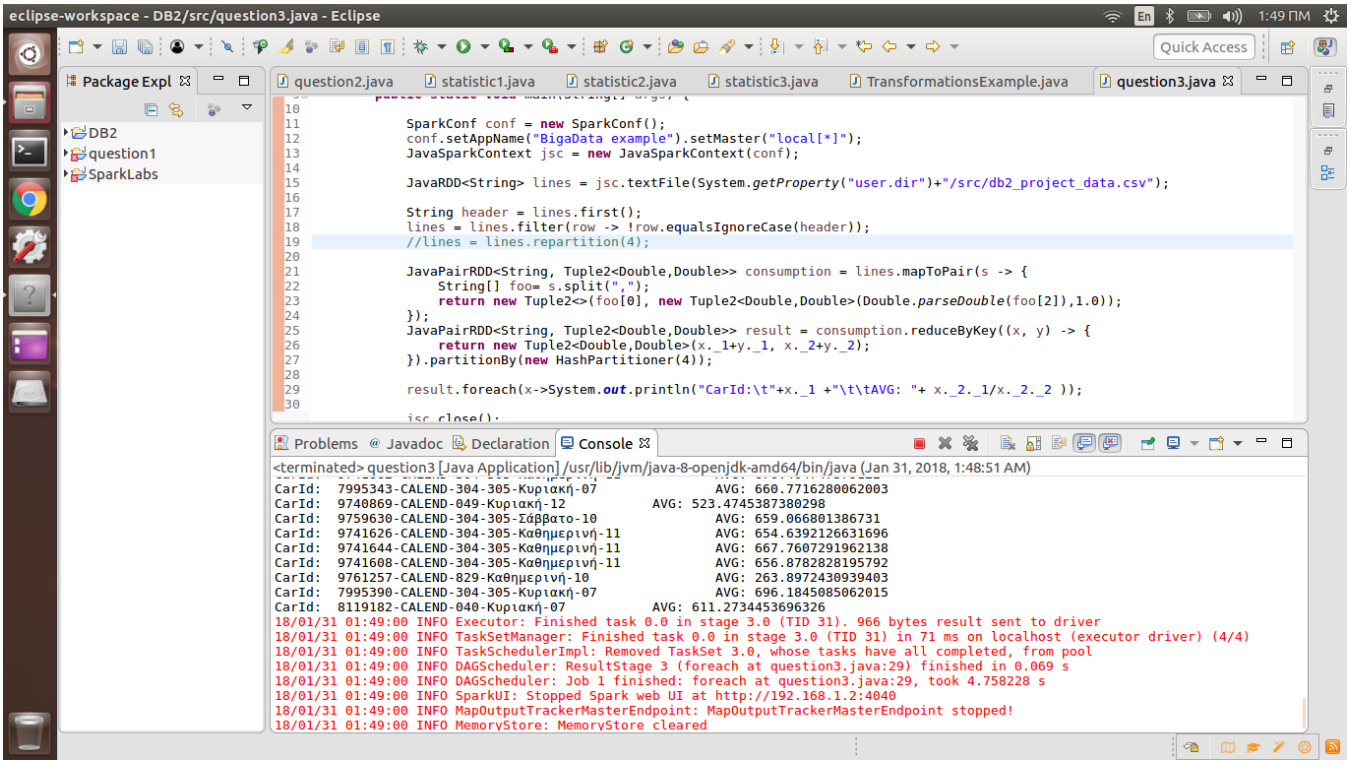
```
18/01/31 01:45:42 INFO TaskSetManager: Finished task 14.0 in stage 2.0 (TID 30) in 21 ms on localhost (executor driver) (14/15)
18/01/31 01:45:42 INFO Executor: Finished task 11.0 in stage 2.0 (TID 27). 1095 bytes result sent to driver
18/01/31 01:45:42 INFO TaskSetManager: Finished task 11.0 in stage 2.0 (TID 27) in 31 ms on localhost (executor driver) (15/15)
18/01/31 01:45:42 INFO TaskSchedulerImpl: Removed TaskSet 2.0, whose tasks have all completed, from pool
18/01/31 01:45:42 INFO DAGScheduler: ResultStage 2 (foreach at statistic3.java:25) finished in 0.145 s
18/01/31 01:45:42 INFO DAGScheduler: Job 1 finished: foreach at statistic3.java:25, took 3.792846 s
18/01/31 01:45:42 INFO SparkUI: Stopped Spark web UI at http://192.168.1.2:4040
18/01/31 01:45:42 INFO MapOutputTrackerMasterEndpoint: MapOutputTrackerMasterEndpoint stopped!
18/01/31 01:45:42 INFO MemoryStore: MemoryStore cleared
18/01/31 01:45:42 INFO BlockManager: BlockManager stopped
18/01/31 01:45:42 INFO BlockManagerMaster: BlockManagerMaster stopped
```

B. Σε αυτό το ερώτημα πήρα σαν βάση το προηγούμενο ερώτημα και το μεταποίησα όπου αυτό χρειάστηκε. Πιο συγκεκριμένα, τα αρχικά βήματα μέχρι το map είναι τα ίδια. Στο map αφού δεν έχω τύπο με τρία στοιχεία έπρεπε να φτιάξω ένα μόνος μου. Οπότε, έβαλα το απλό PairRDD<string, > και στην δεύτερη θέση χρησιμοποίησα το Tuple2 για να μπορέσω να φτιάξω δυο στοιχεία από εκεί που είχα ένα. Έτσι με το mapToPair έβαλα σε κάθε ένα την πρώτη και την τρίτη τιμή κάθε γραμμής και στην τρίτη θέση που δημιούργησα έβαλα έναν άσσο για να μπορέσω μετά να βρω το count των ίδιων γραμμών ώστε να διαιρέσω με αυτό το sum και να βγει ο Μ.Ο.. Τέλος μετά το reduce εμφάνισα τα id και το πηλίκο της προαναφερθείσας διαίρεσης. Παραθέτω Screenshot στην συνέχεια:



C. Σε αυτό το ερώτημα δοκίμασα δυο διαφορετικές προσεγγίσεις. Για αρχή χρησιμοποίησα την εντολή .repartition() και παρατήρησα ότι όσο μεγάλωνα την μεταβλητή της παρένθεσης τόσο ανέβαιναν οι χρόνοι εκτέλεσης. Έτσι στην συνέχεια χρησιμοποίησα το partitionBy(new HashPartitioner(4)) με το οποίο δεν παρατήρησα αισθητές διαφορές από το αρχικό πλάνο πέρα από κάτι μικρές αυξήσεις πάλι του χρόνου. Στο εκτελέσιμο αρχείο υπάρχουν και οι δυο υλοποιήσεις αλλά η μια είναι σε σχόλιο. Παραθέτω Screenshot στην συνέχεια:




eclipse-workspace - DB2/src/question3.java - Eclipse
Quick Access
Package Expl
DB2
question1
SparkLabs
question2.java
statistic1.java
statistic2.java
statistic3.java
TransformationsExample.java
question3.java
SparkConf conf = new SparkConf();
conf.setAppName("BigaData example").setMaster("local[*]");
JavaSparkContext jsc = new JavaSparkContext(conf);
JavaRDD<String> lines = jsc.textFile(System.getProperty("user.dir")+"/src/db2_project_data.csv");
String header = lines.first();
lines = lines.filter(row -> !row.equalsIgnoreCase(header));
//lines = lines.repartition(4);
JavaPairRDD<String, Tuple2<Double,Double>> consumption = lines.mapToPair(s -> {
String[] foo= s.split(",");
return new Tuple2<>(foo[0], new Tuple2<Double,Double>(Double.parseDouble(foo[2]),1.0));
});
JavaPairRDD<String, Tuple2<Double,Double>> result = consumption.reduceByKey((x, y) -> {
return new Tuple2<Double,Double>(x._1+y._1, x._2+y._2);
}).partitionBy(new HashPartitioner(4));
result.foreach(x->System.out.println("CarId:\t"+x._1 +"\t\tAVG: "+ x._2._1/x._2._2));
jsc.close();
Problems
Javadoc
Declaration
Console
<terminated> question3 [Java Application] /usr/lib/jvm/java-8-openjdk-amd64/bin/java (Jan 31, 2018, 1:48:51 AM)
CarId: 7995343-CALEND-304-305-Κυριακή-07 AVG: 660.7716280062003
CarId: 9740869-CALEND-049-Κυριακή-12 AVG: 523.4745387380298
CarId: 9759630-CALEND-304-305-Σάββατο-10 AVG: 659.066801386731
CarId: 9741626-CALEND-304-305-Καθημερινή-11 AVG: 654.6392126631696
CarId: 9741644-CALEND-304-305-Καθημερινή-11 AVG: 667.7607291962138
CarId: 9741608-CALEND-304-305-Καθημερινή-11 AVG: 656.8782828195792
CarId: 9761257-CALEND-829-Καθημερινή-10 AVG: 263.8972430939403
CarId: 7995390-CALEND-304-305-Κυριακή-07 AVG: 696.1845085062015
CarId: 8119182-CALEND-040-Κυριακή-07 AVG: 611.2734453696326
18/01/31 01:49:00 INFO Executor: Finished task 0.0 in stage 3.0 (TID 31). 966 bytes result sent to driver
18/01/31 01:49:00 INFO TaskSetManager: Finished task 0.0 in stage 3.0 (TID 31) in 71 ms on localhost (executor driver) (4/4)
18/01/31 01:49:00 INFO TaskSchedulerImpl: Removed TaskSet 3.0, whose tasks have all completed, from pool
18/01/31 01:49:00 INFO DAGScheduler: ResultStage 3 (foreach at question3.java:29) finished in 0.069 s
18/01/31 01:49:00 INFO DAGScheduler: Job 1 finished: foreach at question3.java:29, took 4.758228 s
18/01/31 01:49:00 INFO SparkUI: Stopped Spark web UI at http://192.168.1.2:4040
18/01/31 01:49:00 INFO MapOutputTrackerMasterEndpoint: MapOutputTrackerMasterEndpoint stopped!
18/01/31 01:49:00 INFO MemoryStore: MemoryStore cleared