

Συστήματα Διαχείρισης Βάσεων Δεδομένων 2018-19

(5ο εξάμηνο)

Ομάδα Εργασίας:p14086, Κουσουννής Κωνσταντίνος



ΠΑΝΕΠΙΣΤΗΜΙΟ ΠΕΙΡΑΙΩΣ

UNIVERSITY OF PIRAEUS

Στοιχεία Φοιτητή

Επώνυμο Φοιτητή Κουσουννής

Όνομα Φοιτητή Κωνσταντίνος

Αριθμός Μητρώου P14086

Email Kwstas654321@gmail.com

Τιτλος Εργασία Συστήματα Διαχείρισης
Βάσεων Δεδομένων 2018-2019

Εργασία Συστήματα Διαχείρισης Βάσεων Δεδομένων 2018-2019

Εισαγωγή

Θα εργαστείτε με ένα σύνολο δεδομένων από την ιστοσελίδα ανοιχτών δεδομένων (Open Data) της κυβέρνησης του Ηνωμένου Βασιλείου, που έχουν δημοσιευθεί από το Υπουργείο Μεταφορών, συγκεκριμένα το: UK Road Safety: Traffic Accidents and Vehicles (URL: <https://www.kaggle.com/tsiaras/uk-road-safety-accidents-andvehicles/home?fbclid=IwAR1fgYHabasR7ObkbySkApilmmDwo7LNs6PQ-7LEgxlpuRkU2x8OUVvIzc>). Το συγκεκριμένο σύνολο δεδομένων αφορά σε αναλυτικές πληροφορίες για τροχαία ατυχήματα και εμπλεκόμενα οχήματα στο Ηνωμένο Βασίλειο κατά τα έτη 2005- 2016. Το υποσύνολο δεδομένων που θα χρησιμοποιήσετε και αποτελείται από δύο αρχεία csv, έχει υποστεί κατάλληλη επεξεργασία, ώστε να εξυπηρετεί τις ανάγκες και τους σκοπούς της εργασίας. Μπορείτε να το κατεβάσετε από τον ακόλουθο σύνδεσμο (URL:

https://www.dropbox.com/s/lzxju6nmcbi8fh/db2_data.zip?dl=0): •

Accident_Information.csv: κάθε γραμμή του αρχείου αντιπροσωπεύει ένα μοναδικό τροχαίο ατύχημα (στήλη Accident_Index). • Vehicle_Information.csv: κάθε γραμμή του αρχείου αντιπροσωπεύει τη συμμετοχή ενός μοναδικού οχήματος σε ένα μοναδικό τροχαίο ατύχημα. Βάσει των παραπάνω, καλείστε να απαντήσετε στα ακόλουθα ερωτήματα:

Ερώτημα 1 (60%). Βελτιστοποίηση Επερωτήσεων σε Κεντρικοποιημένη ΒΔ

Κάθε φορά που εκτελείτε μία από τις παρακάτω επερωτήσεις θα δείχνετε τον χρόνο εκτέλεσης (πάντα θα εκτελείτε την επερώτηση τουλάχιστον δύο φορές και θα κρατάτε τον τελευταίο χρόνο – ο λόγος που δεν κρατάμε απλά τον χρόνο εκτέλεσης της πρώτης φοράς είναι ότι δεν είναι αντιπροσωπευτικός γιατί τα buffers δεν έχουν προλάβει να αρχικοποιηθούν) καθώς και το πλάνο εκτέλεσης (χρησιμοποιώντας την εντολή EXPLAIN, screenshot). Σκοπός είναι κάθε φορά που αλλάζετε κάτι στη ΒΔ, με απώτερο στόχο να βελτιώσετε τους χρόνους εκτέλεσης, να παρατηρείτε αν υπάρχει βελτίωση και πόση είναι αυτή αλλά και να εξηγείτε τη βελτίωση αυτή με βάση τη θεωρία και το πλάνο εκτέλεσης. Η απάντηση στα ερωτήματα a, b, c, d, e πρέπει να γίνει με τη σειρά εμφάνισή τους, δηλαδή, θα απαντήσετε στο ερώτημα b αφού πρώτα έχετε απαντήσει στο a (και ούτω καθεξής), έτσι ώστε οι αλλαγές που κάνατε στο a (π.χ. buffers, parallelism, κτλ.) να συνεχίσουν να είναι ενεργές στο b και έπειτα

(a) Αφού φορτώσετε τα δεδομένα στο Σύστημα Διαχείρισης ΒΔ της PostgreSQL (εντολή "COPY ... WITH CSV HEADER") και ανανεώσετε τα στατιστικά χρησιμοποιώντας την εντολή "VACUUM FULL ...", εκτελέστε τις παρακάτω επερωτήσεις (queries) χρησιμοποιώντας τις

προεπιλεγμένες ρυθμίσεις της PostgreSQL και χωρίς να έχετε δημιουργήσει βοηθητικές δομές (π.χ. ευρετήρια).

Αρχικά κατεβάζουμε τα δεδομένα μας από την ιστοσελίδα που μας δίνεται.

← → C ⌂ Dropbox, Inc [US] | https://www.dropbox.com/s/lzxju6fnmcbl8fh/db2_data.zip?dl=0&fbclid=IwAR3hiSF...
For quick access, place your bookmarks here on the bookmarks bar. Import bookmarks now...

We use cookies so that Dropbox works for you. By using our website, you agree to our use of cookies. [Learn more](#)

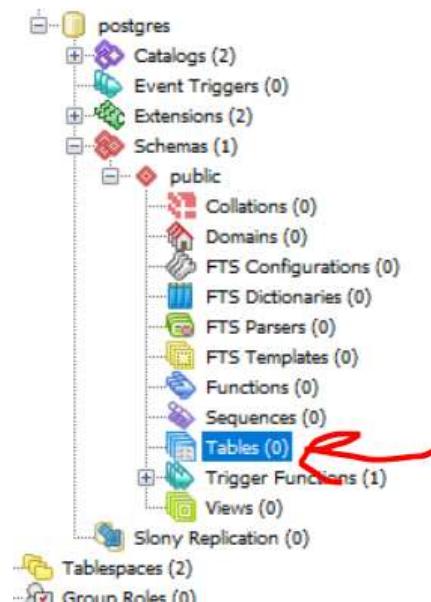
db2_data.zip

Name	Size
db2_Accident_Information.csv	142.52 MB
db2_Vehicle_Information.csv	148.72 MB

VirtualBox-5.2.22-126460-Win	12/17/2018 3:42 A...	Application	111,263 KB
db2_Accident_Information	12/7/2018 9:49 AM	Microsoft Excel Co...	145,941 KB
db2_Vehicle_Information	12/7/2018 9:49 AM	Microsoft Excel Co...	152,287 KB
Windows 10 Pro 64-bit	12/11/2018 5:07 PM	Windows Update	106,020 KB

Τα δύο αρχεία ονομάζονται **db2_Accident_Information, db2_Vehicle_Information**

Αφού κατεβάσουμε τα αρχεία μας ανοίγουμε το pgAdmin3 και δημιουργούμε δυο πίνακες με τα αντίστοιχα χαρακτηριστικά.



Ανοίγουμε ένα query και δίνουμε τις εντολές μας.

Δηλώνουμε τον πίνακα **db2_Accident_Information**

```

create table db2_Accident_Information(
    empty integer,
    Accident_Index varchar(255),
    First_Road_Class varchar(255),
    Accident_Severity varchar(255),
    Date date,
    Urban_or_Rural_Area varchar(255),
    Weather_Conditions varchar(255),
    Year integer,
    InScotland varchar(255),
    primary key(Accident_Index)
);

copy db2_Accident_Information from 'D:\db2_Accident_Information.csv' Delimiter ',' CSV HEADER;

```

Αυτός έχει τις αντιστοιχες στήλες

A	B	C	D	E	F	G	H	I	J
Accident_Index	1st_Road_Class		Accident_Severity	Date	Urban_or_Rural_Area	Weather_Conditions	Year	InScotland	
0 200501B500001	A		Serious	1/4/2005	Urban	Raining no high winds	2005	No	

Κάθε στήλη που έχω ορίσει είναι η αντίστοιχη με την στήλη που έχω μέσα στο csv αρχείο όπως βλέπουμε και παραπάνω.

Κα βάζουμε σαν πρωτεύων κλειδί (**primary key**) την στήλη Accident_Index η οποία δεν έχει κανένα ίδιο στοιχείο σύμφωνα με την εκφώνηση της άσκησης.

- Accident_Information.csv: κάθε γραμμή του αρχείου αντιπροσωπεύει ένα μοναδικό τροχαίο ατύχημα (στήλη Accident_Index).

The screenshot shows the IBM DB2 SQL Editor interface. The SQL Editor pane contains the previously shown SQL code for creating the table and copying data from a CSV file. The Output pane at the bottom displays the message: "Query returned successfully: 1917274 rows affected, 17.6 secs execution time." and "1917274 rows affected." The status bar at the bottom right indicates "DOS Ln 16, Col 2, Ch 409" and "17.6 secs".

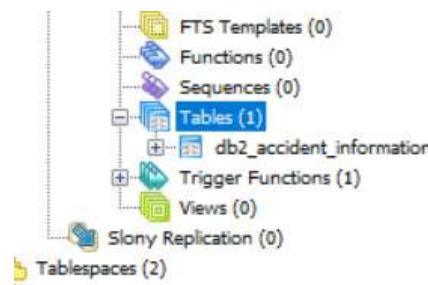
Τα αρχεία μου τα έχω αποθηκευμένα στο path 'D:\db2_Accident_Information'

Τρέχουμε το query και βλέπουμε ότι τελείωσε επιτυχημένα.

Output pane

Data Output Explain Messages History

Query returned successfully: 1917274 rows affected, 17.6 secs execution time.



Πηγαίνουμε μετά στην βάση μας και ελέγχουμε ναν έχουν περαστεί η πίνακες.

Οπως βλέπουμε ο πίνακας έχει δημιουργηθεί κοιτάμε και τα δεδομένα.

File Edit View Tools Help

empty accident_index [PK] character varying(255) first_road_class character varying(255) accident_severity character varying(255) date urban_or_rural_area character varying(255) weather_conditions character varying(255) year integer inscotland character varying(255)

	empty	accident_index [PK]	first_road_class	accident_severity	date	urban_or_rural_area	weather_conditions	year	inscotland
1	0	200501BS00001	A	Serious	2005-01-04	Urban	Raining no high winds	2005	No
2	1	200501BS00002	B	Slight	2005-01-05	Urban	Fine no high winds	2005	No
3	2	200501BS00003	C	Slight	2005-01-06	Urban	Fine no high winds	2005	No
4	3	200501BS00004	A	Slight	2005-01-07	Urban	Fine no high winds	2005	No
5	4	200501BS00005	Unclassified	Slight	2005-01-10	Urban	Fine no high winds	2005	No
6	5	200501BS00006	Unclassified	Slight	2005-01-11	Urban	Raining no high winds	2005	No
7	6	200501BS00007	C	Slight	2005-01-13	Urban	Fine no high winds	2005	No
8	7	200501BS00009	A	Slight	2005-01-14	Urban	Fine no high winds	2005	No
9	8	200501BS00010	A	Slight	2005-01-15	Urban	Fine no high winds	2005	No
10	9	200501BS00011	B	Slight	2005-01-15	Urban	Fine no high winds	2005	No
11	10	200501BS00012	A	Slight	2005-01-16	Urban	Fine no high winds	2005	No
12	11	200501BS00014	A	Slight	2005-01-25	Urban	Fine no high winds	2005	No
13	12	200501BS00015	Unclassified	Slight	2005-01-11	Urban	Raining no high winds	2005	No
14	13	200501BS00016	A	Slight	2005-01-18	Urban	Raining no high winds	2005	No
15	14	200501BS00017	A	Slight	2005-01-18	Urban	Fine no high winds	2005	No
16	15	200501BS00018	A	Slight	2005-01-18	Urban	Fine no high winds	2005	No
17	16	200501BS00019	Unclassified	Serious	2005-01-20	Urban	Fine no high winds	2005	No
18	17	200501BS00020	A	Slight	2005-01-21	Urban	Fine no high winds	2005	No
19	18	200501BS00021	B	Slight	2005-01-21	Urban	Fine no high winds	2005	No
20	19	200501BS00022	A	Serious	2005-01-08	Urban	Fine no high winds	2005	No
21	20	200501BS00023	Unclassified	Slight	2005-01-24	Urban	Fine no high winds	2005	No
22	21	200501BS00024	B	Slight	2005-01-24	Urban	Fine no high winds	2005	No
23	22	200501BS00025	A	Slight	2005-01-24	Urban	Fine no high winds	2005	No
24	23	200501BS00028	C	Slight	2005-01-18	Urban	Fine no high winds	2005	No
25	24	200501BS00029	A	Slight	2005-01-29	Urban	Fine no high winds	2005	No
26	25	200501BS00031	C	Slight	2005-01-19	Urban	Raining no high winds	2005	No

Scratch pad

Όπως βλέπουμε τα δεδομένα μας έχουν περάσει κανονικά.

Ακολουθούμε την ίδια διαδικασία για τον επόμενο πίνακα.

```

SQL Editor | Graphical Query Builder
Previous queries
create table db2_Vehicle_Information(
    empty integer,
    Accident_Index varchar(255),
    Age_Band_of_Driver varchar(255),
    Age_of_Vehicle varchar(255),
    make varchar(255),
    model varchar(255),
    Sex_of_Driver varchar(255),
    Vehicle_Type varchar(255)
);
copy db2_Vehicle_Information from 'D:\db2_Vehicle_Information.csv' Delimiter ',' CSV HEADER encoding 'windows-1251';

```

Δημιουργούμε έναν πίνακα db2_Vehicle_Information

δηλώνουμε τις στήλες μας αντίστοιχα με το db2_Vehicle_Information.csv

A	B	C	D	E	F	G	H	I
1	Accident_Index	Age_Band_of_Driver	Age_of_Vehicle	make	model	Sex_of_Driver	Vehicle_Type	
2	0 200401B500001	26 - 35		3 ROVER	45 CLASSIC 16V	Male		109
3	1 200401B500002	26 - 35		BMW	C1	Male		109

Έχω αντίστοιχα το αρχείο αποθηκευμένο στο path '[D:\db2_Vehicle_Information](#)'

Πατάω play και περνάω τα στοιχεία μου.

```

Query - postgres on localhost:5432*
File Edit Query Favorites Macros View Help
SQL Editor | Graphical Query Builder
Previous queries
create table db2_Vehicle_Information(
    empty integer,
    Accident_Index varchar(255),
    Age_Band_of_Driver varchar(255),
    Age_of_Vehicle varchar(255),
    make varchar(255),
    model varchar(255),
    Sex_of_Driver varchar(255),
    Vehicle_Type varchar(255)
);
copy db2_Vehicle_Information from 'D:\db2_Vehicle_Information.csv' Delimiter ',' CSV HEADER encoding 'windows-1251';

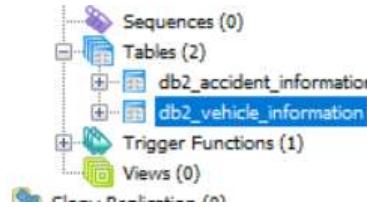
Output pane
Data Output Explain Messages History
Query returned successfully: 2177205 rows affected, 6.9 secs execution time.

2177205 rows affected.

```



Τα δεδομένα μου πέρασαν με επιτυχία.



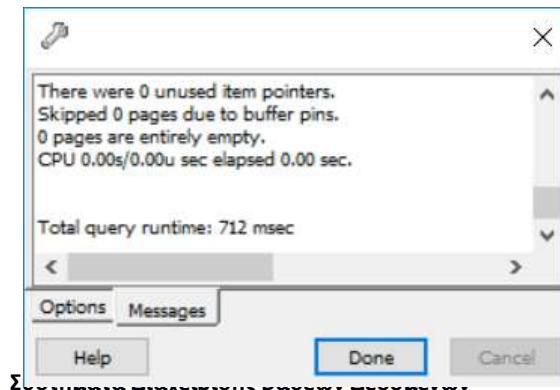
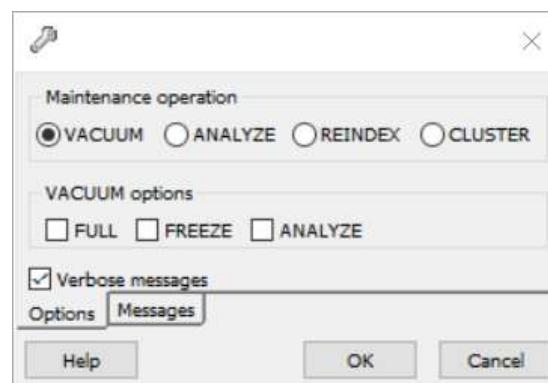
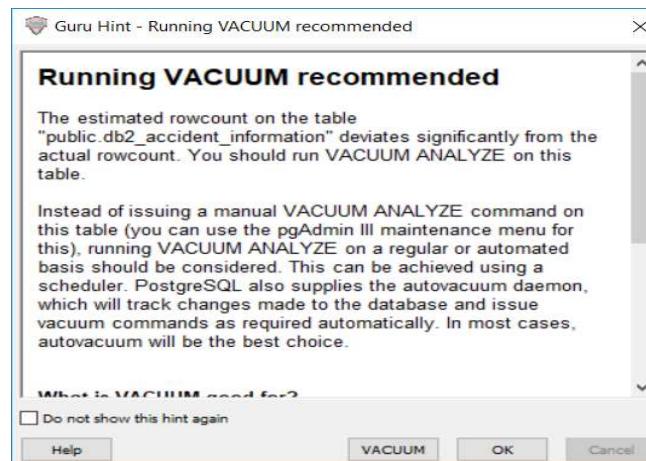
Ο πίνακας έχει οριστεί.

Edit Data - PostgreSQL 9.5 (localhost:5432) - postgres - public.db2_vehicle_information

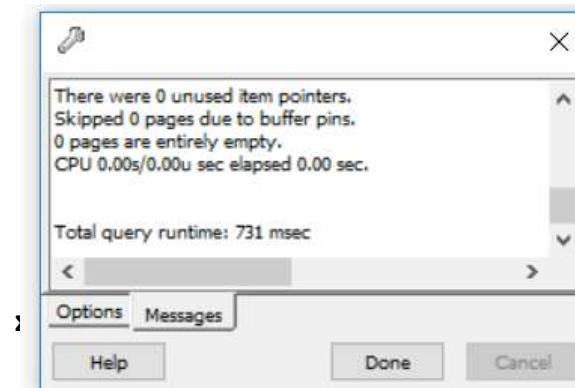
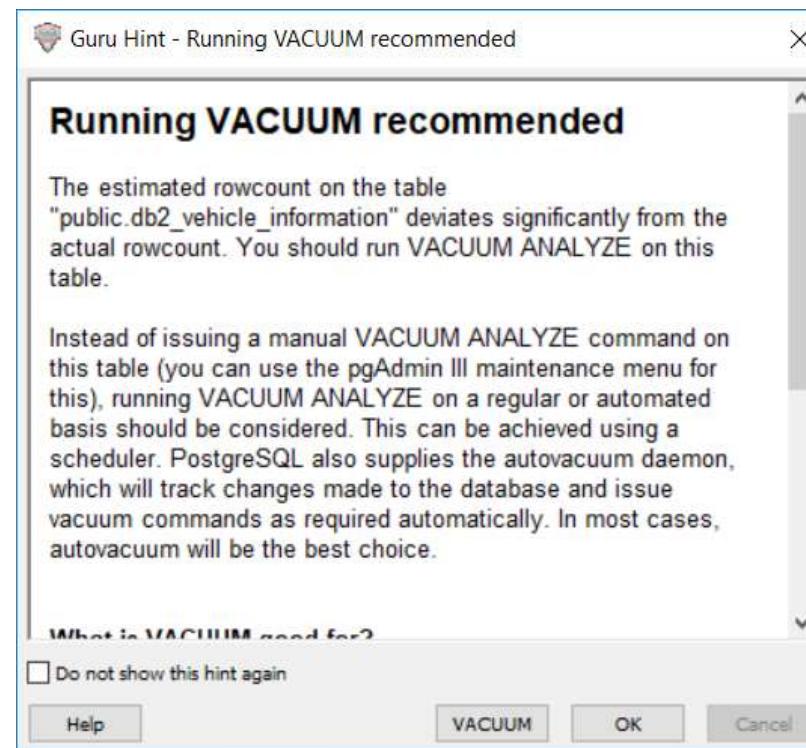
	empty	accident_index	age_band_of_driver	age_of_vehicle	make	model	sex_of_driver	vehicle_type
	integer	character varying(255)						
1	0	200401BS00001	26 - 35	3.0	ROVER	45 CLASSIC 16V	Male	109
2	1	200401BS00002	26 - 35		BMW	C1	Male	109
3	2	200401BS00003	26 - 35	4.0	NISSAN	MICRA CELEBRATION 16V	Male	109
4	3	200401BS00003	66 - 75		LONDON TAXIS INT	TXII GOLD AUTO	Male	109
5	4	200401BS00004	26 - 35	1.0	PIAGGIO	VESPA ET4	Male	Motorcycle 125cc and under
6	5	200401BS00004	36 - 45	10.0	VOLKSWAGEN		Male	109
7	6	200401BS00009	26 - 35		PIAGGIO	VESPA GT 125	Male	Motorcycle 125cc and under
8	7	200401BS00010	36 - 45		BMW	R1100 RT	Male	109
9	8	200401BS00012	46 - 55	3.0	MERCEDES		Male	109
10	9	200401BS00013	26 - 35	4.0	VOLKSWAGEN	GOLF V5	Female	109

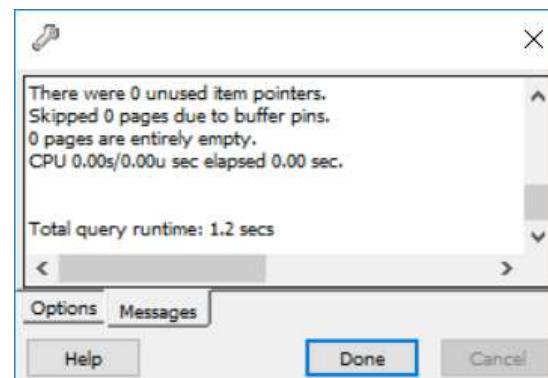
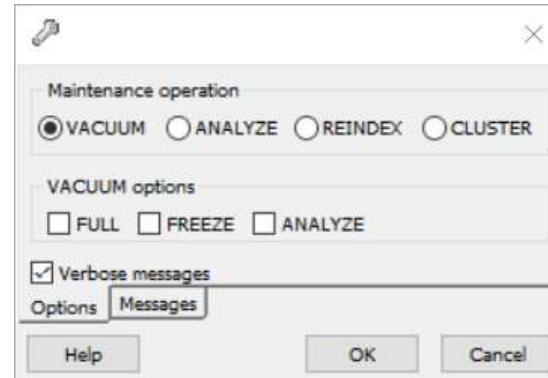
Κάνουμε Vacuum και στους δυο πίνακες

db2_accident_information



db2_vehicle_information





Σε κάθε εκτέλεση κάθε ερωτήματος έχω μια 1η εκτέλεση και μια 2η εκτέλεση εμείς θα παίρνουμε και μας ενδιαφέρει η 2η εκτέλεση(αν και λέω δεύτερη εκτέλεση μπορεί να έχω τρέξει περισσότερες από τρείς φορές την επερώτηση) από την εκφώνηση:

(πάντα θα εκτελείτε την επερώτηση τουλάχιστον δύο φορές και θα κρατάτε τον τελευταίο χρόνο – ο λόγος που δεν κρατάμε απλά τον χρόνο εκτέλεσης της πρώτης φοράς είναι ότι δεν είναι αντιπροσωπευτικός γιατί τα buffers δεν έχουν προλάβει να αρχικοποιηθούν.

- Βρείτε πόσα ατυχήματα έγιναν ανά κατηγορία δρόμου (road class).

Query - postgres@localhost:5432*

File Edit Query Favourites Macros View Help

SQL Editor Graphical Query Builder

Previous queries:

```
select first_road_class, count(accident_index) as Accidents_by_road_class
from db2_accident_information
group by first_road_class |
```

Output pane

first_road_class	accidents_by_road_class
1 Motorway	73641
2 Primary	51215
3 Unclassified	558137
4 C	166972
5 A (H)	5141
6 L	870268

OK DOS Ln 3, Col 27, Ch 132 6 rows. 796 msec

File Edit Query Favourites Macros View Help

SQL Editor Graphical Query Builder

Previous queries:

```
select first_road_class, count(accident_index) as Accidents_by_road_class
from db2_accident_information
group by first_road_class |
```

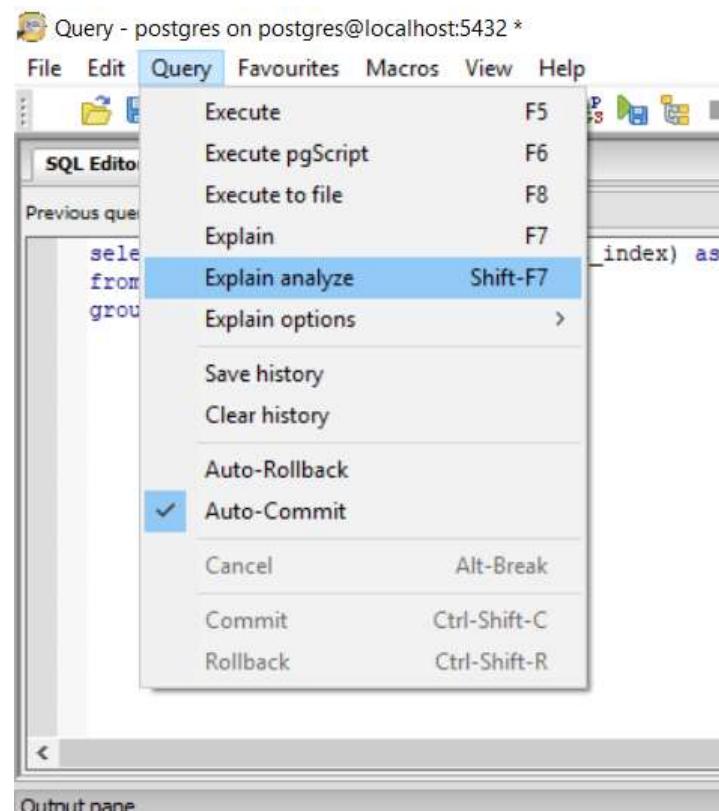
Θέλουμε να μας εμφανίσει τα ατυχήματα που έγιναν ανά κατηγορία δρόμου.

```
select first_road_class,count(accident_index) as Accidents_by_road_class  
from db2_accident_information  
group by first_road_class
```

οπότε με την count μετράμε ποσά είναι τα ατυχήματα αφού τα κάνουμε ομαδοποίηση με βάση τον δρόμο

Στην συνέχεια πατάμε explain query

1η Εκτέλεση



Query - postgres on postgres@localhost:5432 *

File Edit Query Favourites Macros View Help

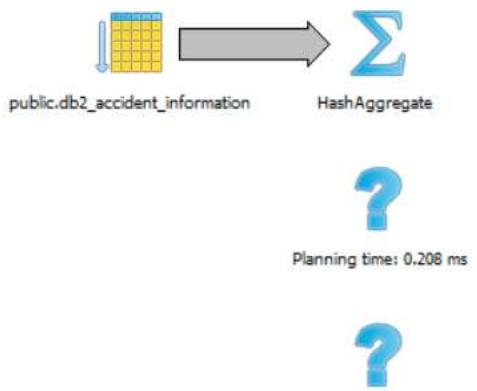
SQL Editor Graphical Query Builder

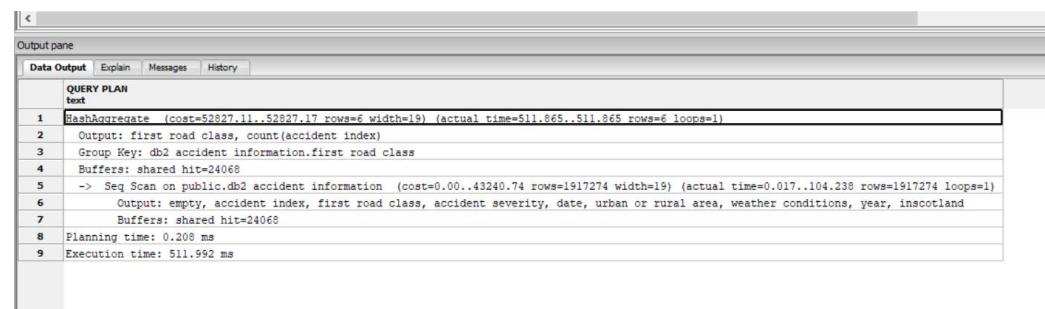
Previous queries

```
select first_road_class, count(accident_index) as Accidents_by_road_class
from db2_accident_information
group by first_road_class |
```

Output pane

Data Output Explain Messages History



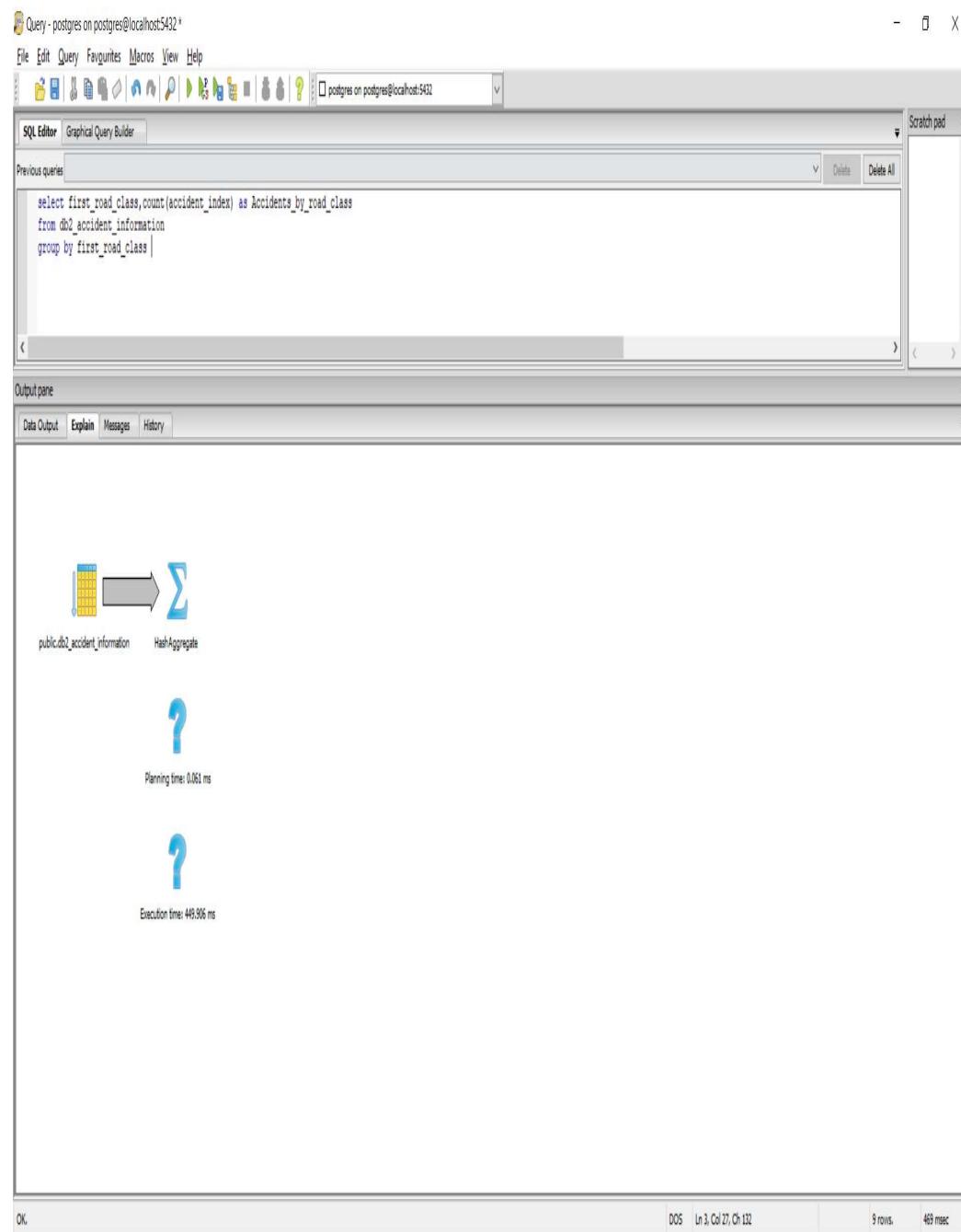


```
1 HashAggregate (cost=52827.11..52827.17 rows=6 width=19) (actual time=511.865..511.865 rows=6 loops=1)
  2   Output: first road class, count(accident index)
  3   Group Key: db2 accident information.first road class
  4   Buffers: shared hit=24068
  5   ->  Seq Scan on public.db2 accident information (cost=0.00..43240.74 rows=1917274 width=18) (actual time=0.017..104.238 rows=1917274 loops=1)
  6     Output: empty, accident index, first road class, accident severity, date, urban or rural area, weather conditions, year, inscotland
  7     Buffers: shared hit=24068
  8 Planning time: 0.208 ms
  9 Execution time: 511.992 ms
```

Planning time:0.208ms

Execution time:511.992ms

2η Εκτέλεση(αυτή που μας ενδιαφέρει και κρατάμε(αν και λέω δεύτερη εκτέλεση μπορεί να έχω τρέξει περισσότερες από τρείς φορές την επερώτηση))



The screenshot shows a PostgreSQL graphical query builder interface. The top menu bar includes File, Edit, Query, Favorites, Macros, View, Help, and a connection dropdown set to "postgres on postgres@localhost:5432". Below the menu is a toolbar with various icons. The main area has tabs for SQL Editor and Graphical Query Builder, with the latter selected. A "Previous queries" list contains a single entry:

```
select first_road_class, count(accident_index) as Accidents_by_road_class
from db2_accident_information
group by first_road_class |
```

The "Output pane" below shows the execution plan:

```
public.db2_accident_information HashAggregate
?
Planning time: 0.061 ms
?
Execution time: 449.306 ms
```

The bottom status bar indicates DOS Ln 3, Col 27, Ch 132, 9 rows, and 463 msec.

Query - postgres on postgres@localhost:5432 *

File Edit Query Favourites Macros View Help

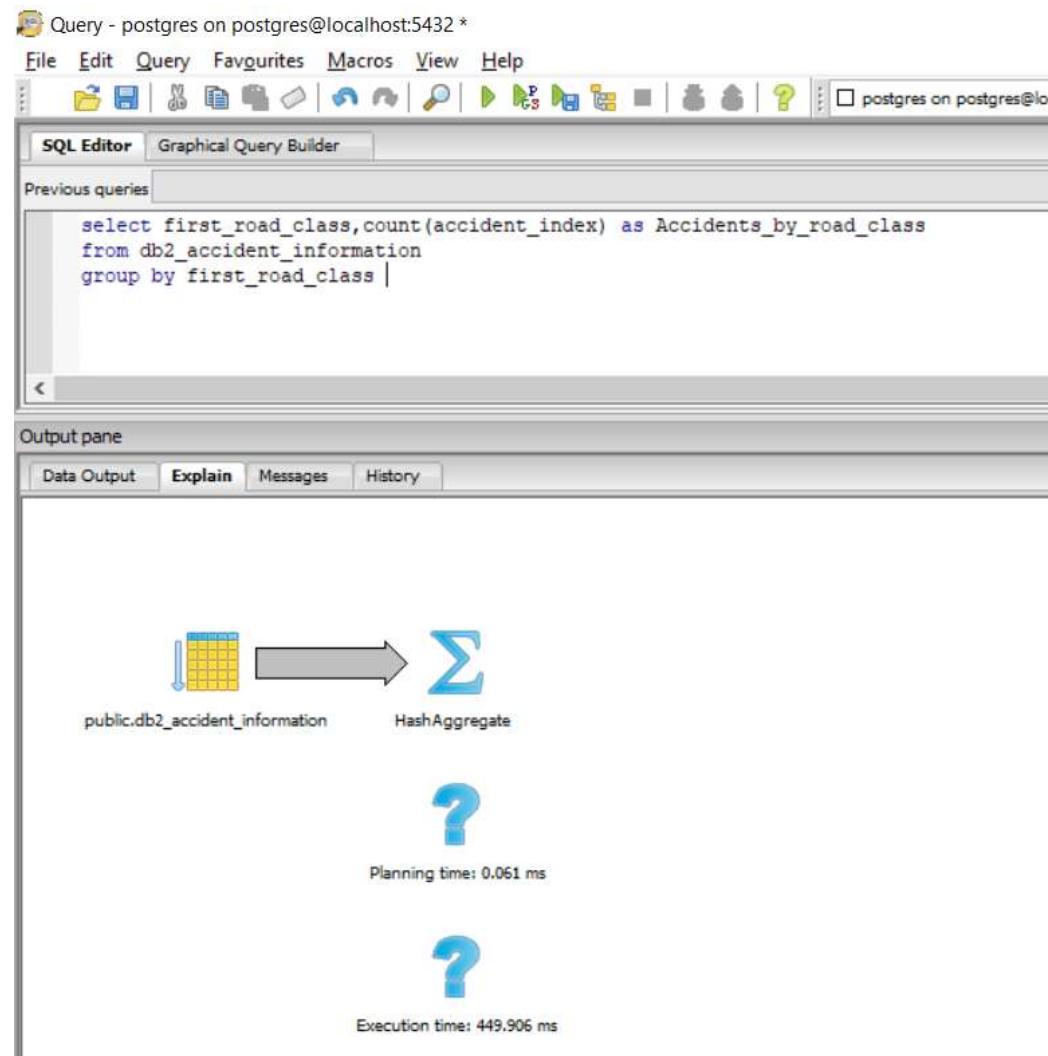
SQL Editor Graphical Query Builder

Previous queries

```
select first_road_class, count(accident_index) as Accidents_by_road_class
from db2_accident_information
group by first_road_class |
```

Output pane

Data Output Explain Messages History



public.db2_accident_information → HashAggregate

Planning time: 0.061 ms

Execution time: 449.906 ms

Output pane

Data Output		Explain	Messages	History
QUERY PLAN				
text				
1	HashAggregate (cost=52827.11..52827.17 rows=6 width=19) (actual time=449.873..449.874 rows=6 loops=1)			
2	Output: first road class, count(accident index)			
3	Group Key: db2 accident information.first road class			
4	Buffers: shared hit=24068			
5	-> Seq Scan on public.db2 accident information (cost=0.00..43240.74 rows=1917274 width=19) (actual time=0.004..92.200 rows=1917274 loops=1)			
6	Output: empty, accident index, first road class, accident severity, date, urban or rural area, weather conditions, year, inscotland			
7	Buffers: shared hit=24068			
8	Planning time: 0.061 ms			
9	Execution time: 449.906 ms			

Planning time:0.061ms

Execution time:449.906ms

time:469msecs

ii. Πόσα είναι τα ατυχήματα ανά κατηγορία δρόμου και ανά κατηγορία ατυχήματος (Accident Severity).

6	Output: empty, accident index
7	Buffers: shared hit=24068
8	Planning time: 0.061 ms
9	Execution time: 449.906 ms

Query - postgres on postgres@localhost:5432*

File Edit Query Favorites Macros View Help

SQL Editor Graphical Query Builder

Previous queries

```
select first_road_class,accident_severity,count(accident_index) as countAccidents_by_road_class_and_accidentseverity
from db2_accident_information
group by first_road_class,accident_severity;
```

Scratch pad X

Output pane

	first_road_class	accident_severity	countaccidents_by_road_class_and_accidentseverity
1	A (H)	Slight	4429
2	Motorway	Serious	7509
3	Unclassified	Fatal	4043
4	A	Slight	736960
5	B	Serious	36619
6	B	Slight	203066
7	A	Serious	118492
8	C	Fatal	2010
9	Motorway	Slight	64847
10	A (H)	Serious	603
11	Motorway	Fatal	125
12	Unclassified	Serious	76472
13	C	Slight	141852
14	B	Fatal	3430
15	A (H)	Fatal	109
16	C	Serious	23110
17	A	Fatal	13818
18	Unclassified	Slight	477622

Data Output Explain Messages History

DOS Ln 3, Col 44, Ch 193 18 rows. 609 msec

Query - postgres on postgres@localhost:5432 *

File Edit Query Favorites Macros View Help

SQL Editor Graphical Query Builder

Previous queries

```
select first_road_class,accident_severity,count(accident_index) as countAccidents_by_road_class_and_Accidentseverity
from db2_accident_information
group by first_road_class,accident_severity;
```

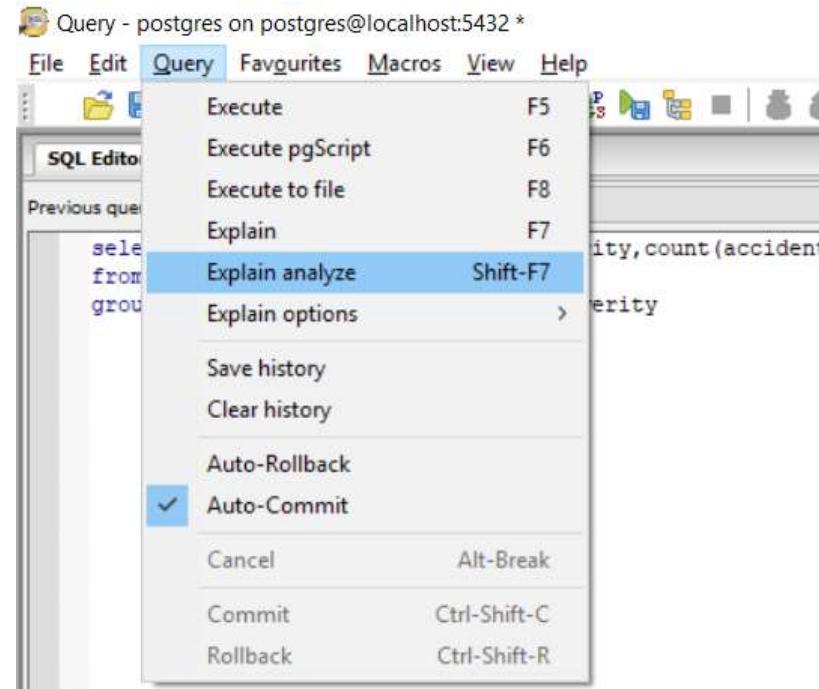
Output pane

	first_road_class character varying(255)	accident_severity character varying(255)	countaccidents_by_road_class_and_accidentseverity bigint
1	A(M)	Slight	4429
2	Motorway	Serious	7509
3	Unclassified	Fatal	4043
4	A	Slight	736960
5	B	Serious	36619
6	B	Slight	203066
7	A	Serious	119492
8	C	Fatal	2010
9	Motorway	Slight	64847
10	A(M)	Serious	603
11	Motorway	Fatal	1285
12	Unclassified	Serious	76472
13	C	Slight	141852
14	B	Fatal	3430
15	A(M)	Fatal	109
16	C	Serious	23110
17	A	Fatal	13816

```
Query - postgres on postgres@localhost:5432 *
File Edit Query Favurites Macros View Help
SQL Editor Graphical Query Builder
Previous queries
select first_road_class,accident_severity,count(accident_index) as countAccidents_by_road_class_and_Accidentseverity
from db2_accident_information
group by first_road_class,accident_severity
```

Θέλω να βρω πόσα είναι τα ατυχήματα ανά κατηγορία δρόμου και ανά κατηγορία ατυχήματος (Accident Severity).

Κάνω select για να εμφανίσω τον δρόμο την σοβαρότητα του ατυχήματος και πόσα είναι αυτά τα ατυχήματα και τα ομαδοποιώ με βάση τον δρόμο και την σοβαρότητα.



1η εκτέλεση

Query - postgres on postgres@localhost:5432 *

File Edit Query Favourites Macros View Help

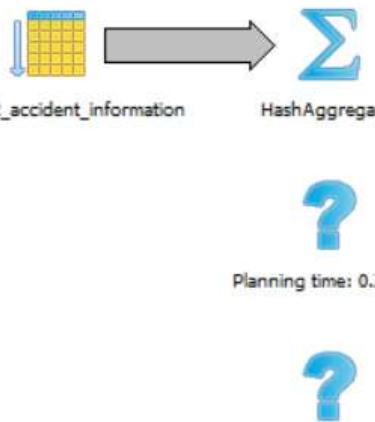
SQL Editor Graphical Query Builder

Previous queries

```
select first_road_class,accident_severity,count(accident_index
from db2_accident_information
group by first_road_class,accident_severity|
```

Output pane

Data Output Explain Messages History



```
Output pane
Data Output Explain Messages History
QUERY PLAN
text
1 HashAggregate (cost=57620.30..57620.48 rows=18 width=26) (actual time=642.834..642.836 rows=18 loops=1)
2   Output: first road class, accident severity, count(accident index)
3   Group Key: db2 accident information.first road class, db2 accident information.accident severity
4   Buffers: shared hit=24068
5   -> Seq Scan on public.db2 accident information (cost=0.00..43240.74 rows=1917274 width=26) (actual time=0.019..103.737 rows=1917274 loops=1)
6     Output: empty, accident index, first road class, accident severity, date, urban or rural area, weather conditions, year, inscotland
7     Buffers: shared hit=24068
8 Planning time: 0.233 ms
9 Execution time: 642.954 ms
```

2η Εκτέλεση(αυτή που μας ενδιαφέρει και κρατάμε(αν και λέω δεύτερη εκτέλεση μπορεί να έχω τρέξει περισσότερες από τρείς φορές την επερώτηση))

Query - postgres on postgres@localhost:5432 *

File Edit Query Favourites Macros View Help

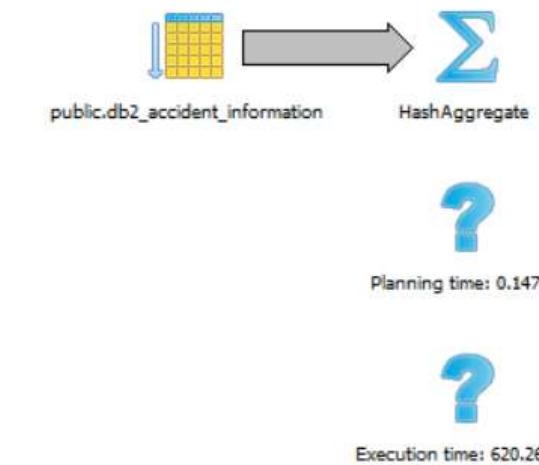
SQL Editor Graphical Query Builder

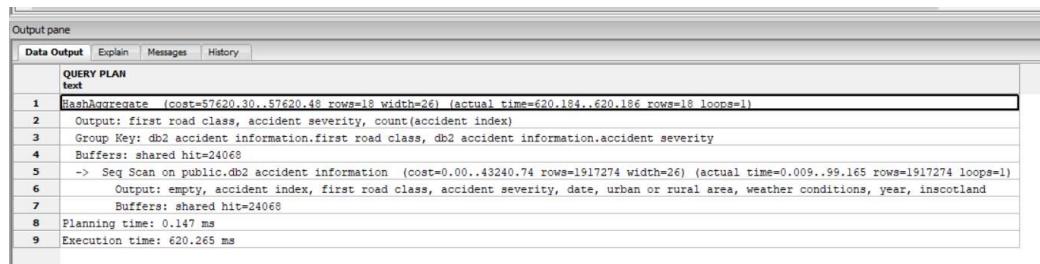
Previous queries

```
select first_road_class, accident_severity, count(accident_index)
from db2_accident_information
group by first_road_class, accident_severity|
```

Output pane

Data Output Explain Messages History





```

Output pane
Data Output Explain Messages History
QUERY PLAN
text
1 HashAggregate (cost=57620.30..57620.48 rows=18 width=26) (actual time=620.184..620.186 rows=18 loops=1)
  2   Output: first road class, accident severity, count(accident index)
  3   Group Key: db2 accident information.first road class, db2 accident information.accident severity
  4   Buffers: shared hit=24068
  5   -> Seq Scan on public.db2 accident information (cost=0.00..43240.74 rows=1917274 width=26) (actual time=0.009..99.165 rows=1917274 loops=1)
  6     Output: empty, accident index, first road class, accident severity, date, urban or rural area, weather conditions, year, inscotland
  7     Buffers: shared hit=24068
  8 Planning time: 0.147 ms
  9 Execution time: 620.265 ms
  
```

Planning time:0.147ms

Execution time:620.265ms

time:641msecs

iii. Βρείτε πόσα ατυχήματα έγιναν σε αστική περιοχή πριν από την 1/1/2010 και η ηλικιακή κατηγορία του οδηγού είναι 26-35.

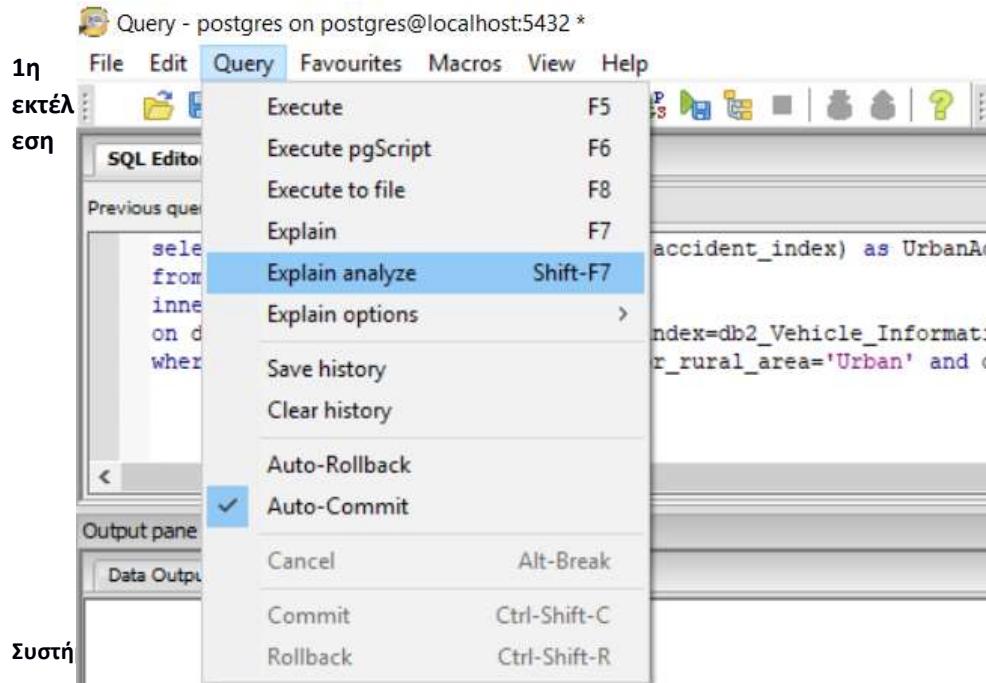


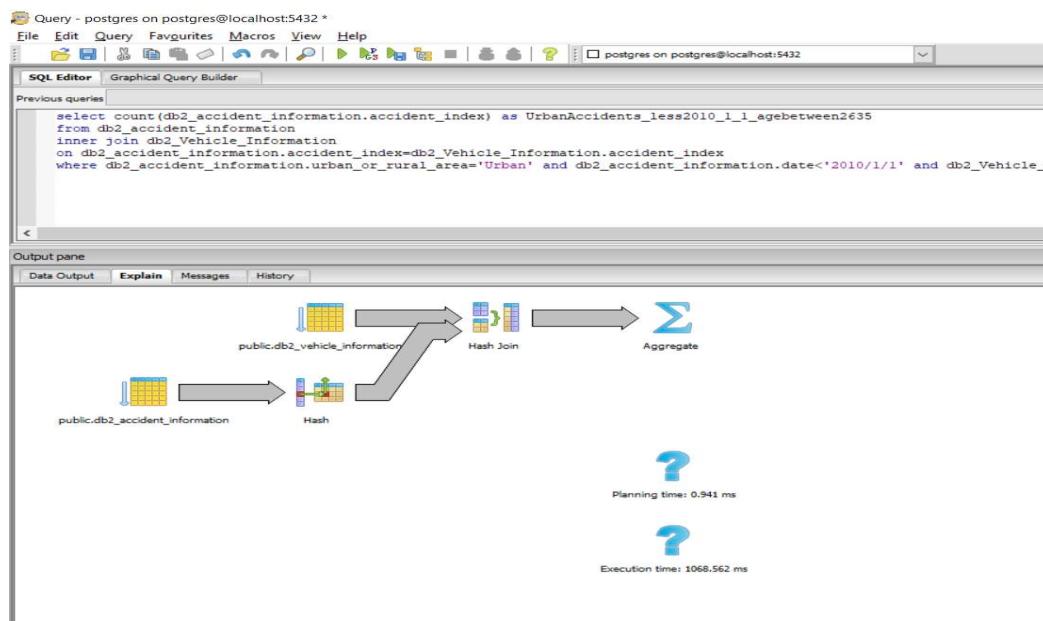
```

Query - postgres on localhost:5432 *
File Edit Query Favourites Macros View Help
postgres on localhost:5432
SQL Editor | Graphical Query Builder
Previous queries
select count(db2_accident_information.accident_index) as UrbanAccidents_less2010_1_1_agebetween2635
from db2_accident_information
inner join db2_Vehicle_Information
on db2_accident_information.accident_index=db2_Vehicle_Information.accident_index
where db2_accident_information.urban_or_rural_area='Urban' and db2_accident_information.date<'2010/1/1' and db2_Vehicle_Information.age_band_of_driver='26 - 35'
Output pane
Data Output Explain Messages History
urbanaccidents_less2010_1_1_agebetween2635
bigint
1 86298
  
```

```
Query - postgres on postgres@localhost:5432 *
File Edit Query Favourites Macros View Help
[...]
SQL Editor Graphical Query Builder
Previous queries
select count(db2_accident_information.accident_index) as UrbanAccidents_less2010_1_1_agebetween2635
from db2_accident_information
inner join db2_Vehicle_Information
on db2_accident_information.accident_index=db2_Vehicle_Information.accident_index
where db2_accident_information.urban_or_rural_area='Urban' and db2_accident_information.date<'2010/1/1' and db2_Vehicle_Information.age_band_of_driver='26 - 35'
```

Επιλέγουμε πόσα αυτοχήματα έγιναν σε αστική περιοχή οπότε κάνουμε count την στήλη accident_index αλλά επειδή θέλουμε να είναι σε αστική περιοχή πριν από την 1/1/2010 και η ηλικιακή κατηγορία του οδηγού είναι 26-35 ενώνουμε τους δυο πίνακες με inner join για να κάνουμε ελέγχους από τους πίνακες db2_accident_information, db2_Vehicle_information και τέλος βάζουμε την συνθήκη για τους περιορισμούς μας **where**
db2_accident_information.urban_or_rural_area='Urban' and
db2_accident_information.date<'2010/1/1' and
db2_Vehicle_Information.age_band_of_driver='26 – 35'





Output pane	
	Data Output
1	Aggregate (cost=127243.84..127243.85 rows=1 width=14) (actual time=1067.385..1067.389 rows=1 loops=1)
2	Output: count(db2 accident information.accident index)
3	Buffers: shared hit=50687, temp read=4031 written=4001
4	-> Hash Join (cost=62835.40..126906.45 rows=134956 width=14) (actual time=410.424..1061.901 rows=86298 loops=1)
5	Output: db2 accident information.accident index
6	Hash Cond: ((db2 vehicle information.accident index)::text = (db2 accident information.accident index)::text)
7	Buffers: shared hit=50687, temp read=4031 written=4001
8	-> Seq Scan on public.db2 vehicle information (cost=0.00..53834.06 rows=449448 width=14) (actual time=0.009..287.774 rows=450531 loops=1)
9	Output: db2 vehicle information.accident index
10	Filter: ((db2 vehicle information.age band of driver)::text = '26 - 35'::text)
11	Rows Removed by Filter: 1726674
12	Buffers: shared hit=26619
13	-> Hash (cost=52827.11..52827.11 rows=575703 width=14) (actual time=383.183..383.183 rows=573888 loops=1)
14	Output: db2 accident information.accident index
15	Buckets: 16 Memory Usage: 2645kB
16	Buffers: shared hit=24065, temp written=2223
17	-> Seq Scan on public.db2 accident information (cost=0.00..52827.11 rows=575703 width=14) (actual time=0.007..287.355 rows=573888 loops=1)
18	Output: db2 accident information.accident index
19	Filter: ((db2 accident information.date < '2010-01-01'::date) AND ((db2 accident information.urban or rural area)::text = 'Urban'::text))
20	Rows Removed by Filter: 1343386
21	Buffers: shared hit=24068
22	Planning time: 0.941 ms
23	Execution time: 1068.562 ms

Planning time:0.941ms

Execution time:1068.562ms

2η Εκτέλεση(αυτή που μας ενδιαφέρει (αν και λέω δεύτερη εκτέλεση μπορεί να έχω τρέξει περισσότερες από τρείς φορές την επερώτηση))

Query - postgres on postgres@localhost:5432 *

File Edit Query Favorites Macros View Help

SQL Editor Graphical Query Builder

Previous queries

```
select count(db2_accident_information.accident_index) as UrbanAccidents_less2010_1_1_agebetween2635
from db2_accident_information
inner join db2_Vehicle_Information
on db2_accident_information.accident_index=db2_Vehicle_Information.accident_index
where db2_accident_information.urban_or_rural_area='Urban' and db2_accident_information.date<'2010/1/1' a
```

Output pane

Data Output Explain Messages History

Planning time: 0.977 ms

Execution time: 1061.916 ms

Output pane

Data Output Explain Messages History

QUERY PLAN

```
1 Aggregate (cost=27243.84..137243.85 rows=1 width=14) (actual time=1061.377..1061.377 rows=1 loops=1)
  2   Output: count(db2 accident information.accident index)
  3   Buffers: shared hit=50687, temp read=4031 written=4001
  4   -> Hash Join (cost=62835.40..126906.45 rows=134956 width=14) (actual time=399.349..1055.787 rows=86298 loops=1)
      5     Output: db2 accident information.accident index
      6     Hash Cond: ((db2 vehicle information.accident index)::text = (db2 accident information.accident index)::text)
      7     Buffers: shared hit=50687, temp read=4031 written=4001
      8     -> Seq Scan on public.db2 vehicle information (cost=0.00..53834.06 rows=449448 width=14) (actual time=0.019..294.868 rows=450531 loops=1)
      9       Output: db2 vehicle information.accident index
      10      Filter: ((db2 vehicle information.age band of driver)::text = '26 - 35'::text)
      11      Rows Removed by Filter: 1726674
      12      Buffers: shared hit=26619
      13      -> Hash (cost=52827.11..52827.11 rows=575703 width=14) (actual time=372.512..372.512 rows=573888 loops=1)
      14        Output: db2 accident information.accident index
      15        Buckets: 131072  Batches: 16  Memory Usage: 2645kB
      16        Buffers: shared hit=24068, temp written=2223
      17        -> Seq Scan on public.db2 accident information (cost=0.00..52827.11 rows=575703 width=14) (actual time=0.008..255.364 rows=573888 loops=1)
      18          Output: db2 accident information.accident index
      19          Filter: ((db2 accident information.date < '2010-01-01'::date) AND ((db2 accident information.urban or rural area)::text = 'Urban'::text))
      20          Rows Removed by Filter: 1343386
      21          Buffers: shared hit=24068
  22 Planning time: 0.977 ms
  23 Execution time: 1061.916 ms
```

OK DOS Ln 5. Col 161. Ch 412

Planning time:0.977ms

Execution time:1061.916ms

time:1msecs

iv. Βρείτε πόσα ατυχήματα έγιναν σε αγροτική περιοχή το έτος 2012 και η ηλικιακή κατηγορία του οδηγού είναι 36-45.

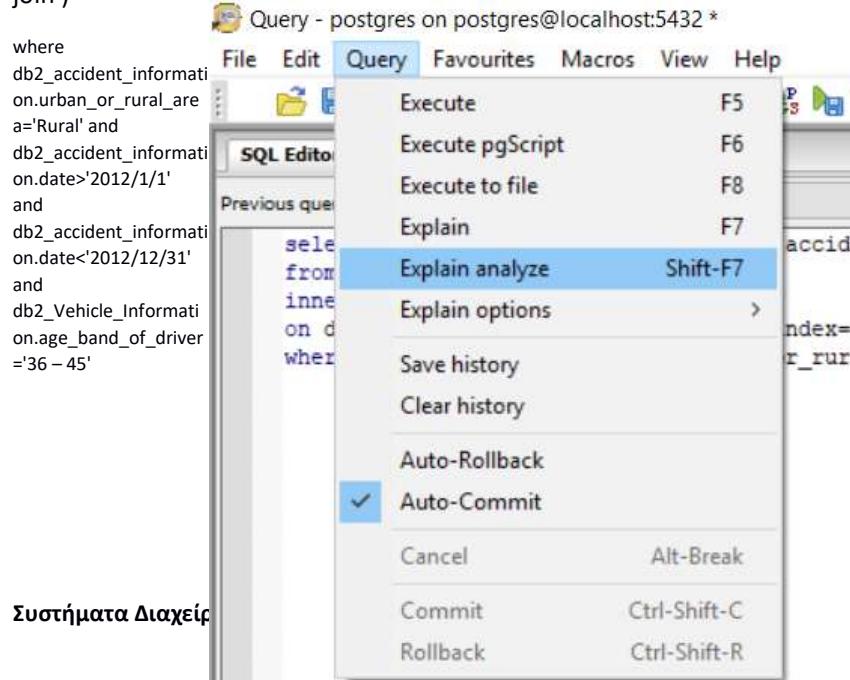


```
Query - postgres on postgres@localhost:5432 *
File Edit Query Favourites Macros View Help
SQL Editor Graphical Query Builder
Previous queries
select count(db2_accident_information.accident_index) as RuralAccidents_Year2012_agebetween3645
from db2_accident_information
inner join db2_Vehicle_Information
on db2_accident_information.accident_index=db2_Vehicle_Information.accident_index
where db2_accident_information.urban_or_rural_area='Rural' and db2_accident_information.date>'2012/1/1' and
db2_accident_information.date<'2012/12/31' and db2_Vehicle_Information.age_band_of_driver='36 - 45'

Output pane
Data Output Explain Messages History
ruralaccidents_year2012_agebetween3645
1 12535
```

```
select count(db2_accident_information.accident_index) as RuralAccidents_Year2012_agebetween3645
from db2_accident_information
inner join db2_Vehicle_Information
on db2_accident_information.accident_index=db2_Vehicle_Information.accident_index
where db2_accident_information.urban_or_rural_area='Rural' and db2_accident_information.date>'2012/1/1' and
db2_accident_information.date<'2012/12/31' and db2_Vehicle_Information.age_band_of_driver='36 - 45'
```

Κάνουμε select για να εμφανίσουμε πόσα ατυχήματα έγιναν σε αγροτική περιοχή το έτος 2012 και η ηλικιακή κατηγορία του οδηγού είναι 36-45. Κανουμε σύνδεση πινάκων (inner join)



1η Εκτέλεση

SQL Editor | Graphical Query Builder

Previous queries

```

    it(db2_accident_information.accident_index) as RuralAccidents_Year2012_agebetween3645
    :incident_information
    db2_Vehicle_Information
    .dent_information.accident_index=db2_Vehicle_Information.accident_index
    :incident_information.urban_or_rural_area='Rural' and db2_accident_information.date>'2012/1/1' and db2_accident_inf
  
```

Output pane

Data Output | Explain | Messages | History

The diagram illustrates the execution plan for the query. It starts with two input tables: 'public.db2_vehicle_information' and 'public.db2_accident_information'. The 'public.db2_vehicle_information' table undergoes a 'Hash' operation, represented by a grid with a green dot indicating the hash key. This hashed table then participates in a 'Hash Join' operation with the second table. The 'Hash Join' is labeled 'Hash Join'. The result of the join then undergoes an 'Aggregate' operation, represented by a large blue sigma symbol, to produce the final result.

Planning time: 1.016 ms

Execution time: 575.523 ms

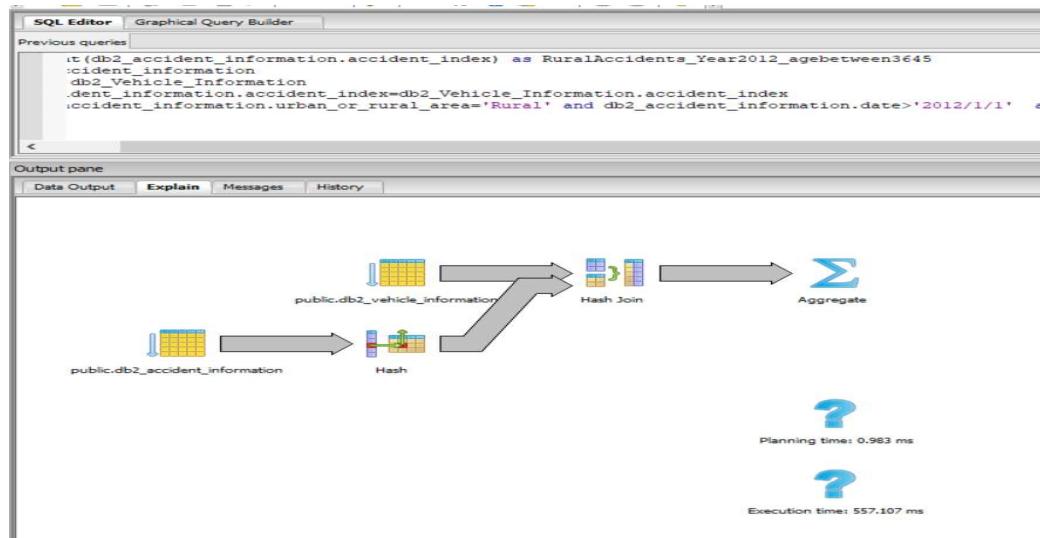
```

Output pane
Data Output Explain Messages History
QUERY PLAN
Node:
1 Aggregate (cost=113909.41..113909.42 rows=1 width=14) (actual time=574.900..574.900 rows=1 loops=1)
  2   Output: count(db2 accident information.accident index)
  3     Buffers: shared hit=50687
  4   -> Hash Join (cost=58276.48..113879.32 rows=12036 width=14) (actual time=416.662..574.141 rows=12535 loops=1)
  5     Output: db2 accident information.accident index
  6     Hash Cond: db2 accident information.accident index::text = (db2 accident information.accident index)::text
  7     Buffers: shared hit=50687
  8     -> Seq Scan on public.db2 vehicle information (cost=0.00..53834.04 rows=435978 width=14) (actual time=0.01..364.198 rows=435986 loops=1)
  9       Output: db2 vehicle information.empty, db2 vehicle information.accident index, db2 vehicle information.age band of driver, db2 vehicle information.age of vehicle, db2 vehicle information.make, db2
 10      Filter: ((db2 vehicle information.age band of driver)::text = '36 - 45';:text)
 11      Rows Removed by Filter: 1741519
 12     Buffers: shared hit=26619
 13     -> Hash Join (cost=57620.30..57620.30 rows=52495 width=14) (actual time=241.410..241.410 rows=50066 loops=1)
 14       Output: db2 accident information.accident index
 15       Buckets: 65536  Batches: 1  Memory Usage: 27624B
 16     Buffers: shared hit=24068
 17     -> Seq Scan on public.db2 accident information (cost=0.00..57620.30 rows=52495 width=14) (actual time=149.744..232.834 rows=50066 loops=1)
 18       Output: db2 accident information.accident index
 19       Filter: ((db2 accident information.date > '2012-01-01';:date) AND (db2 accident information.date < '2012-12-31';:date) AND ((db2 accident information.urban or rural area)::text = 'Rural';:text))
 20       Rows Removed by Filter: 167208
 21     Buffers: shared hit=24068
 22 Planning time: 1.016 ms
 23 Execution time: 575.523 ms

```

OK. DOS Ln 5, Col 209, Ch 456 23 rows. 593 msec

2η Εκτέλεση(αυτή που μας ενδιαφέρει και κρατάμε(αν και λέω δεύτερη εκτέλεση μπορεί να έχω τρέξει περισσότερες από τρείς φορές την επερώτηση))



```

Output pane
Data Output Explain Messages History
QUERY PLAN
1  Aggregate (cost=113809.41..113809.42 rows=1 width=14) (actual time=556.522..556.522 rows=1 loops=1)
2    Output: db2 accident information.accident index
3    Buffers: shared hit=50687
4   -> Hash Join (cost=55276.45..113879.32 rows=12036 width=14) (actual time=409.618..555.811 rows=12535 loops=1)
5     Output: db2 accident information.accident index
6     Hash Cond: ((db2 vehicle information.accident index)::text = (db2 accident information.accident index)::text)
7     Buffers: shared hit=50687
8     -> Seq Scan on public.db2 vehicle information (cost=0.00..53834.06 rows=439978 width=14) (actual time=0.012..249.711 rows=435866 loops=1)
9       Output: db2 vehicle information.empty, db2 vehicle information.accident index, db2 vehicle information.age band of driver, db2 vehicle information.make, db2
10      vehicle information.model, db2 vehicle information.year
11      Filter: ((db2 vehicle information.age band of driver)::text = '36 - 45)::text)
12      Rows Removed by Filter: 1741519
13      Buffers: shared hit=26619
14   -> Hash (cost=57620.30..57620.30 rows=52495 width=14) (actual time=242.632..242.632 rows=50066 loops=1)
15     Output: db2 accident information.accident index
16     Buckets: 65536 Batches: 1 Memory Usage: 2762kB
17     Buffers: shared hit=24068
18   -> Seq Scan on db2 accident information (cost=0.00..57620.30 rows=52495 width=14) (actual time=151.349..234.094 rows=50066 loops=1)
19     Output: db2 accident information.accident index
20     Filter: ((db2 accident information.date > '2012-01-01'::date) AND (db2 accident information.date < '2012-12-31'::date) AND ((db2 accident information.urban or rural area)::text = 'Rural'::text))
21     Rows Removed by Filter: 1847208
22   Buffers: shared hit=24068
23 Planning time: 0.983 ms
23 Execution time: 557.107 ms
  
```

DOS Ln 5, Col 209, Ch 496 23 rows. 578 msec

Planning time:0.983ms

Execution time:557.107ms

time:578msecs

v. Βρείτε τον κατασκευαστή του οχήματος που έχει τα περισσότερα ατυχήματα μεταξύ των ετών 2010 και 2012, η ηλικιακή κατηγορία των οδηγών είναι 26- 35 και η κατηγορία δρόμου είναι A.

```

File Edit Query Favorites Macros View Help
SQL Editor Graphical Query Builder
Scratches
Previous queries
select db2_vehicle_information.make as manufacturer, count(db2_vehicle_information.make) maxAccidents
from db2_vehicle_information
inner join db2_accident_information
on db2_vehicle_information.accident_index=db2_accident_information.accident_index
where db2_accident_information.date>='2010/1/1' and db2_accident_information.date<='2012/12/31' and db2_vehicle_information.Age_Band_of_Driver='26 - 35' and db2_accident_information.first_road_class='A'
group by db2_vehicle_information.make
having count(db2_vehicle_information.make)=(
select max(count)
from(select db2_vehicle_information.make as manufacturer, count(db2_vehicle_information.make)as count
from db2_vehicle_information
inner join db2_accident_information
on db2_vehicle_information.accident_index=db2_accident_information.accident_index
where db2_accident_information.date>='2010/1/1' and db2_accident_information.date<='2012/12/31' and db2_vehicle_information.Age_Band_of_Driver='26 - 35' and db2_accident_information.first_road_class='A'
group by manufacturer) as manufacturerype);
--briskw tous kataskeuyantes kai pose atuximata exoun kanei kai ethn synxelia barw to iso(having count(db2_vehicle_information.make)=)
--gia na brw to megisto apo ayton toyz arisimous pou metrisa ka] to briskw me to na ksema dimourghw ton idio pinaka kataskeuyantes kai pose atuximata gia nia brw parw se ayton ton
--pinaka polo einai to megisto miai kai se agreeption den se afhei na emfaniseis alle stilles an den tis omdopoiheis dhliyah ebrisika to megisto alla emfanise mono ton arisimo
--mono toy kai osi ton kataskeuasti opote me basi to max poy emw briskw messa apo thn ethn count to megisto me tin isotita kai emfanizw apo dipla tou ton kataskeuasti

```

manufacturer	maxaccidents
FORD	6469

```

select db2_vehicle_information.make as manufacturer, count(db2_vehicle_information.make) maxAccidents
from db2_vehicle_information
inner join db2_accident_information
on db2_vehicle_information.accident_index=db2_accident_information.accident_index
where db2_accident_information.date>='2010/1/1' and db2_accident_information.date<='2012/12/31' and
db2_vehicle_information.Age_Band_of_Driver='26 - 35' and db2_accident_information.first_road_class='A'
group by db2_vehicle_information.make
having count(db2_vehicle_information.make)=(
select max(count)
from(select db2_vehicle_information.make as manufacturer, count(db2_vehicle_information.make)as count
from db2_vehicle_information
inner join db2_accident_information
on db2_vehicle_information.accident_index=db2_accident_information.accident_index
where db2_accident_information.date>='2010/1/1' and db2_accident_information.date<='2012/12/31' and db2_vehicle_information.Age_Band_of_Driver='26 - 35' and db2_accident_information.first_road_class='A'
group by manufacturer) as manufacturerype);

```

```
where db2_accident_information.date>='2010/1/1' and db2_accident_information.date<='2012/12/31' and
db2_vehicle_information.Age_Band_of_Driver='26 - 35' and db2_accident_information.first_road_class='A'

group by manufacturer) as manufacturer_type);
```

Βρίσκω τους κατασκευαστές και πόσα ατυχήματα έχουν κάνει. Επιλέγοντας από τους δύο πίνακες κατασκευαστή (**make**) και μετρώντας πόσες φορές εμφανίζεται ο κατασκευαστής με **count(make)** κάνοντας σύνδεση στους δύο πίνακες γιατί θέλω να είναι τα ατυχήματα μεταξύ των ετών 2010 και 2012, η ηλικιακή κατηγορία των οδηγών είναι 26- 35 και η κατηγορία δρόμου είναι A ομαδοποιώντας με βάση τον κατασκευαστή **group by make** άρα

```
select db2_vehicle_information.make as manufacturer, count(db2_vehicle_information.make) maxAccidents
from db2_vehicle_information
inner join db2_accident_information
on db2_vehicle_information.accident_index=db2_accident_information.accident_index
where db2_accident_information.date>='2010/1/1' and db2_accident_information.date<='2012/12/31' and
db2_vehicle_information.Age_Band_of_Driver='26 - 35' and db2_accident_information.first_road_class='A'

group by db2_vehicle_information.make
```

και στην συνέχεια βάζω το ίσο(having count(db2_vehicle_information.make)=) από την στήλη με τους αριθμούς που είδη έχω βρει και το συγκρίνω με το μέγιστο που βρισκω στην συνέχεια δηλαδή

```
having count(db2_vehicle_information.make)=(select max(count))
```

για να βρω το μέγιστο από αυτούς τους αριθμούς και το βρίσκω με το να ξανά δημιουργήσω όπως έκανα και στην αρχή τον ίδιο πίνακα κατασκευαστές και πόσα ατυχήματα έκανε ο κάθε κατασκευαστής με τους ίδιους περιορισμού και την ίδια ομαδοποίηση για να βρω πάνω σε αυτόν τον πίνακα ποιος είναι ο μέγιστος αριθμός μιας και σε aggregation δεν σε αφήνει να εμφανίσεις άλλες στήλες αν δεν τις ομαδοποιήσεις . Δηλαδή έβρισκα τον μέγιστο αλλά εμφανίζεις μόνο τον αριθμό μόνο του και όχι και τον κατασκευαστή οπότε με βάση το μέγιστο (max) που βρίσκω μέσα από την στήλη count βρίσκω το μέγιστο και μετά με βάση την ισότητα που έχω βάλει παραπάνω having count(db2_vehicle_information.make)=(select max(count)) βρίσκω από την αρχική στήλη count το μέγιστο και εμφανίζω από δίπλα του τον κατασκευαστή.

```
having count(db2_vehicle_information.make)=(select max(count))
from(select db2_vehicle_information.make as manufacturer, count(db2_vehicle_information.make)as count
from db2_vehicle_information
inner join db2_accident_information
on db2_vehicle_information.accident_index=db2_accident_information.accident_index
where db2_accident_information.date>='2010/1/1' and db2_accident_information.date<='2012/12/31' and
db2_vehicle_information.Age_Band_of_Driver='26 - 35' and db2_accident_information.first_road_class='A'

group by manufacturer) as manufacturer_type);
```

Βλέπουμε τον μέγιστο που εμφανίζει

Query - postgres on postgres@localhost:5432 *

File Edit Query Favourites Macros View Help

SQL Editor Graphical Query Builder

Scratchpad

Previous queries:

```

select db2_vehicle_information.make as manufacturer, count(db2_vehicle_information.make) maxAccidents
from db2_vehicle_information
inner join db2_accident_information
on db2_vehicle_information.vehicle_index=db2_accident_information.accident_index
where db2_accident_information.date>='2010/1/1' and db2_accident_information.date<='2012/12/31' and db2_vehicle_information.Age_Band_of_Driver='26 - 35' and db2_accident_information.first_road_class
group by db2_vehicle_information.make
having count(db2_vehicle_information.make)=(
    select max(maxAccidents)
    from(select db2_vehicle_information.make as manufacturer, count(db2_vehicle_information.make)as count
        from db2_vehicle_information
        inner join db2_accident_information
        on db2_vehicle_information.vehicle_index=db2_accident_information.accident_index
        where db2_accident_information.date>='2010/1/1' and db2_accident_information.date<='2012/12/31' and db2_vehicle_information.Age_Band_of_Driver='26 - 35' and db2_accident_information.first_road_class
        group by manufacturer) as manufacturer
)
```

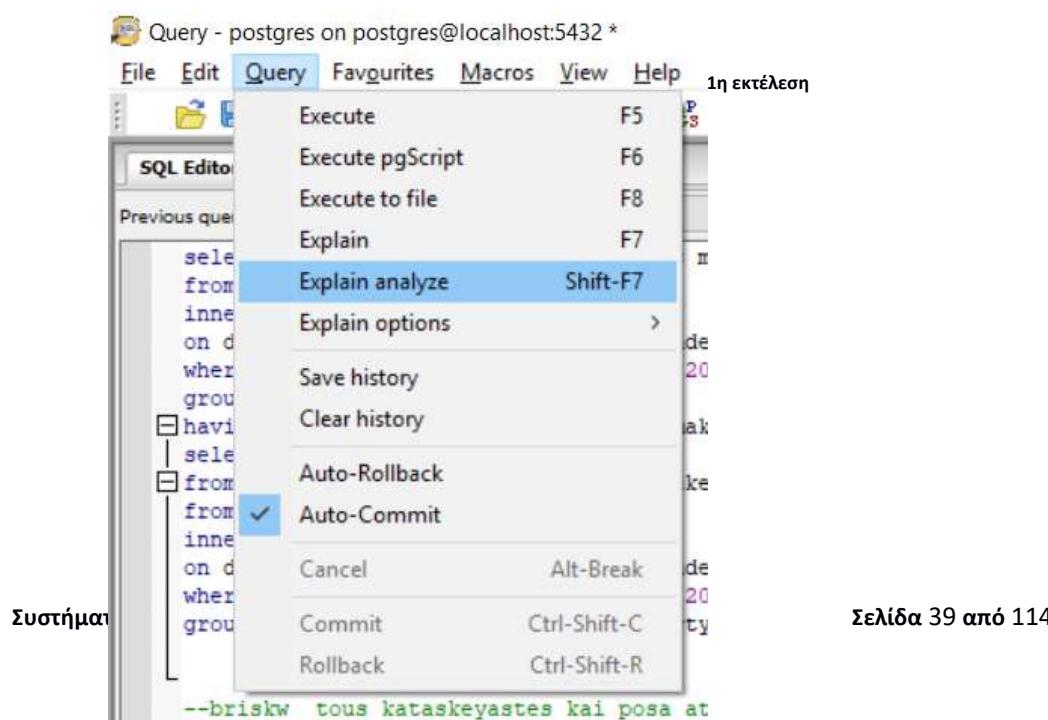
--briskw tous kataskeyastes kai posa atuximate exoun kanei kai synxeseis barw to iso(having count(db2_vehicle_information.make))
--gia na brw to megisto apo ayton toy arithmos pou metise kai to briskw me to na kana dimorfouw ton idio pinaka kataskeyastes kai posa arithmata gia na brw posw se ayron ton
--arithmo pou eishai to megisto mias kai se apereigont den se afinei na emfaniseis allies stiles an den tis meadoupoliseis dikhia chrisha to megisto me tin leitoti kai emfanize mono ton arithmo
--mono toy kai oti ton kataskeusastis opote me hasei to max posw seaytis apo tis sthlii count to megisto me tin leitoti kai emfanize apo diplo tou ton kataskeusasth

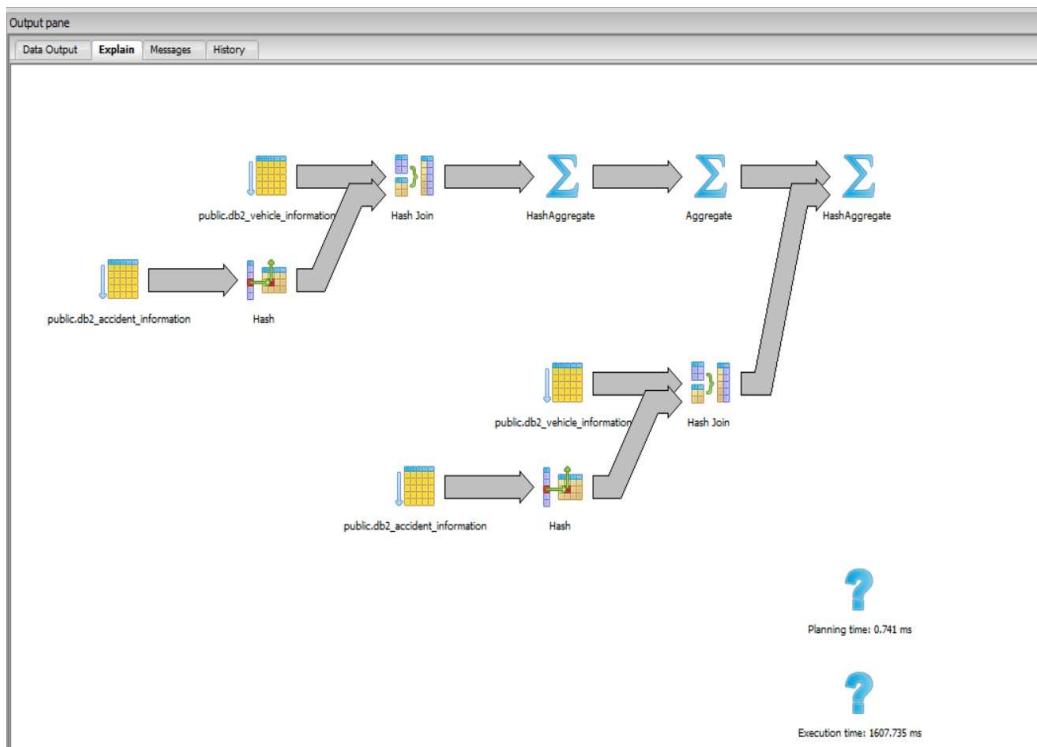
Output pane

manufacturer	maxaccidents
FORD	6469

OK DOS Lr 18, Col 65, Ch 1267 1 row... 1.9 secs

Ο καλύτερος κατασκευαστής είναι ο **FORD 6469** ατυχήματα





Planning time: 0.741 ms



Execution time: 1607.735 ms

Output pane

Data Output	Explain	Messages	History
-------------	---------	----------	---------

```

QUERY PLAN
text
1 HashAggregate (cost=12724.86..247625.48 rows=50 width=7) (actual time=1606.472..1606.477 rows=1 loops=1)
  2 HashJoin (db2 vehicle information.make, count(db2 vehicle information.make))
    3 Group By: db2 vehicle information.make
    4 Filter: (count(db2 vehicle information.make) = $0)
    5 Rows Removed by Filter: 216
    6 Buffers: shared hit=101374, temp read=4692 written=4690
    7 InitPlan 1 (returns 60)
    8   -> Aggregate (cost=123752.26..123752.27 rows=1 width=0) (actual time=797.274..797.274 rows=1 loops=1)
        9     Output: max(count(db2 vehicle information 1.make))
        10    Buffers: shared hit=50697, temp read=2346 written=2340
        11   -> HashAggregate (cost=123752.26..123752.27 rows=1 width=7) (actual time=797.227..797.256 rows=1 loops=1)
        12     Group By: db2 vehicle information 1.make
        13     Output: max(count(db2 vehicle information 1.make))
        14     Buffers: shared hit=50697, temp read=2346 written=2340
        15   -> Hash Join (cost=61222.90..123508.32 rows=48593 width=7) (actual time=429.235..786.349 rows=53584 loops=1)
            16     Output: db2 vehicle information 1.make
            17       Hash Cond: (db2 vehicle information 1.accident index)::text = (db2 accident information 1.accident index)::text
            18       Buffers: shared hit=50697, temp read=2346 written=2340
            19     -> Seq Scan on public.db2 vehicle information db2 vehicle information 1 (cost=0..0.53834.06 rows=449448 width=21) (actual time=0.009..300.780 rows=450531 loops=1)
            20       Output: db2 vehicle information 1.make, db2 vehicle information 1.accident index
            21       Filter: (db2 vehicle information 1.age band of driver)::text = '26 - 35'::text
            22       Rows Removed by Filter: 1726674
            23       Buffers: shared hit=26619
            24     -> Hash (cost=57620.30..57620.30 rows=207248 width=14) (actual time=278.971..278.971 rows=207329 loops=1)
            25       Output: db2 accident information 1.accident index
            26       Buckets: 131072 Batches: 4 Memory Usage: 3355kB
            27       Buffers: shared hit=24068, temp written=444
            28     -> Seq Scan on public.db2 accident information db2 accident information 1 (cost=0..0.57620.30 rows=207248 width=14) (actual time=55.278..235.226 rows=207329 loops=1)
            29       Output: db2 accident information 1.accident index
            30       Filter: ((db2 accident information 1.date >= '2010-01-01'::date) AND (db2 accident information 1.date <= '2012-12-31'::date) AND ((db2 accident information 1.first road class)::text = 'A'::text))
            31       Rows Removed by Filter: 1709945
            32       Buffers: shared hit=24068
  <

```

DOS Ln 20. Col 190. Ch 1740 52 rows. 1.6 secs

Output pane

Data Output	Explain	Messages	History
-------------	---------	----------	---------

```

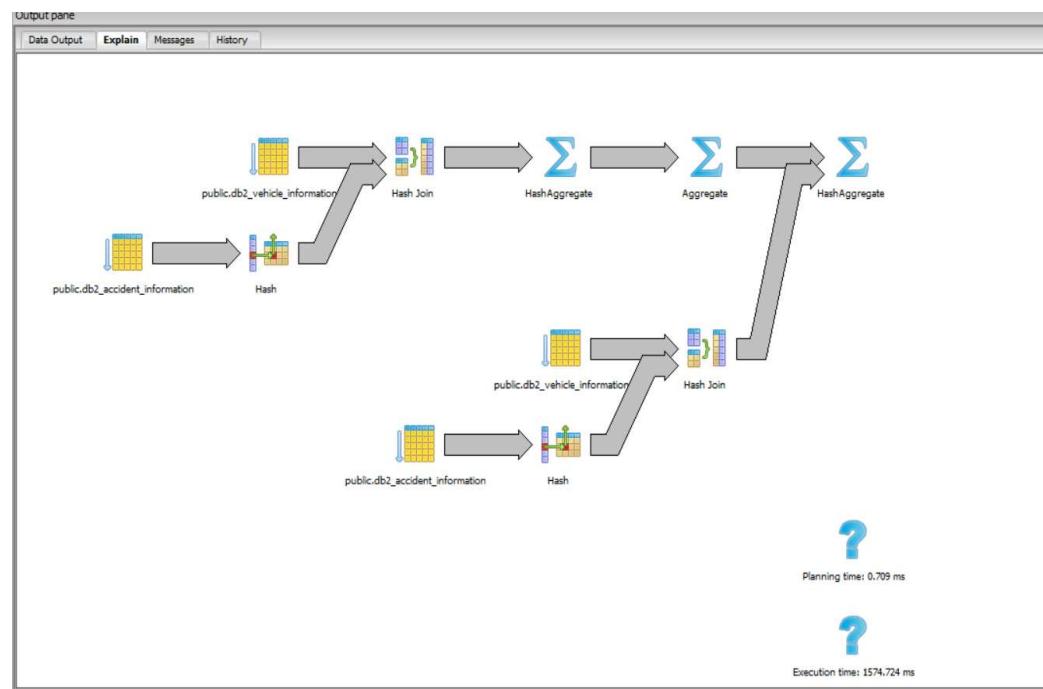
QUERY PLAN
text
22 Rows Removed by Filter: 1726674
23 Buffers: shared hit=26619
24 -> Hash (cost=57620.30..57620.30 rows=207248 width=14) (actual time=278.971..278.971 rows=207329 loops=1)
25   Output: db2 accident information 1.accident index
26   Buckets: 131072 Batches: 4 Memory Usage: 3355kB
27   Buffers: shared hit=24068, temp written=444
28 -> Seq Scan on public.db2 accident information db2 accident information 1 (cost=0..0.57620.30 rows=207248 width=14) (actual time=55.278..235.226 rows=207329 loops=1)
29   Output: db2 accident information 1.accident index
30   Filter: ((db2 accident information 1.date >= '2010-01-01'::date) AND (db2 accident information 1.date <= '2012-12-31'::date) AND ((db2 accident information 1.first road class)::text = 'A'::text))
31   Rows Removed by Filter: 1709945
32   Buffers: shared hit=24068
33 -> Hash Join (cost=61222.90..123508.32 rows=48593 width=7) (actual time=432.694..798.120 rows=53584 loops=1)
34   Output: db2 vehicle information 1.accident index::text = (db2 accident information.accident index)::text
35   Hash Join (db2 vehicle information.accident index)::text = (db2 accident information.accident index)::text
36   Buffers: shared hit=50697, temp read=2340
37   -> Seq Scan on public.db2 vehicle information (cost=0..0.53834.06 rows=449448 width=21) (actual time=0.005..306.606 rows=450531 loops=1)
38   Output: db2 vehicle information.make, db2 vehicle information.accident index
39   Filter: ((db2 vehicle information.age band of driver)::text = '26 - 35'::text)
40   Rows Removed by Filter: 1726674
41   Buffers: shared hit=26619
42 -> Hash (cost=57620.30..57620.30 rows=207248 width=14) (actual time=253.443..283.443 rows=207329 loops=1)
43   Output: db2 accident information.accident index
44   Buckets: 131072 Batches: 4 Memory Usage: 3355kB
45   Buffers: shared hit=24068, temp written=444
46 -> Seq Scan on public.db2 accident information (cost=0..0.57620.30 rows=207248 width=14) (actual time=86.621..238.373 rows=207329 loops=1)
47   Output: db2 accident information.accident index
48   Filter: ((db2 accident information.date >= '2010-01-01'::date) AND (db2 accident information.date <= '2012-12-31'::date) AND ((db2 accident information.first road class)::text = 'A'::text))
49   Rows Removed by Filter: 1709945
50   Buffers: shared hit=24068
51 Planning time: 0.743 ms
52 Execution time: 1607.735 ms
<
```

DOS Ln 20. Col 190. Ch 1740 52 rows. 1.6 secs

Planning time:0.741 ms

Execution time:1607.735ms

2η Εκτέλεση(αυτή που μας ενδιαφέρει και κρατάμε(αν και λέω δεύτερη εκτέλεση μπορεί να έχω τρέξει περισσότερες από τρείς φορές την επερώτηση))



Planning time 0.709ms

Execution time:1574.724ms

time:1.5secs

?

Planning time: 0.709 ms

?

Execution time: 1574.724 ms

```

Output pane
Data Output Explain Messages History
QUERY PLAN
text
1 HashAggregate (cost=247624.86..247625.48 rows=50 width=7) (actual time=1573.586..1573.591 rows=1 loops=1)
  Output: db2 vehicle information.make, count(db2 vehicle information.make)
  Group Key: db2 vehicle information.make
  Filter: (count(db2 vehicle information.make) = 0)
  Rows Removed by Filter: 216
  Buffers: shared hit=101374, temp read=4692 written=4680
  InitPlan 1 (returns 20)
    --> Aggregate (cost=123752.26..123752.27 rows=1 width=8) (actual time=786.158..786.158 rows=1 loops=1)
      Output: max(count(db2 vehicle information.l.make))
      Buffers: shared hit=50687, temp read=2346 written=2340
      --> HashAggregate (cost=123751.13..123751.69 rows=50 width=7) (actual time=786.112..786.140 rows=217 loops=1)
        Output: db2 vehicle information.l.make, count(db2 vehicle information.l.make)
        Group Key: db2 vehicle information.l.make
        Filter: (count(db2 vehicle information.l.make) = 0)
        Rows Removed by Filter: 216
        Buffers: shared hit=101374, temp read=4692 written=4680
        InitPlan 1 (returns 20)
          --> Aggregate (cost=123751.13..123751.13 rows=1 width=8) (actual time=786.158..786.158 rows=1 loops=1)
            Output: max(count(db2 vehicle information.l.make))
            Buffers: shared hit=50687, temp read=2346 written=2340
            --> HashAggregate (cost=123751.13..123751.69 rows=50 width=7) (actual time=786.112..786.140 rows=217 loops=1)
              Output: db2 vehicle information.l.make, count(db2 vehicle information.l.make)
              Group Key: db2 vehicle information.l.make
              Filter: (count(db2 vehicle information.l.make) = 0)
              Rows Removed by Filter: 216
              Buffers: shared hit=101374, temp read=4692 written=4680
              InitPlan 1 (returns 20)
                --> Hash Join (cost=419.928..419.928 rows=53584 loops=1)
                  Output: db2 vehicle information.l.make
                  Hash Cond: (db2 vehicle information.l.accident index)::text = (db2 accident information 1.accident index)
                  Buffers: shared hit=50687, temp read=2346 written=2340
                  --> Seq Scan on public.db2 vehicle information db2 vehicle information l (cost=0.0..53534.06 rows=449448 width=21) (actual time=0.009..258.775 rows=450531 loops=1)
                  Output: db2 vehicle information.l.make, db2 vehicle information.l.accident index
                  Filter: ((db2 vehicle information.l.age band of driver)::text = '26 - 35'::text)
                  Rows Removed by Filter: 1726674
                  Buffers: shared hit=26619
                  --> Hash (cost=57620.30..57620.30 rows=207248 width=14) (actual time=274.108..274.108 rows=207329 loops=1)
                  Output: db2 accident information.accident index
                  Buckets: 131072 Batches: 1 Memory Usage: 3355kB
                  Buffers: shared hit=4068, temp written=44
                  --> Seq Scan on public.db2 accident information db2 accident information 1 (cost=0.0..57620.30 rows=207248 width=14) (actual time=0.546..231.597 rows=207329 loops=1)
                  Output: db2 accident information 1.accident index
                  Filter: ((db2 accident information 1.date >= '2010-01-01'::date) AND (db2 accident information 1.date <= '2012-12-31'::date) AND ((db2 accident information 1.first road class)::text = 'A'::text))
                  Rows Removed by Filter: 1709945
                <
OK.                               DOS  In 20. Col 190. Ch 1740      52 rows.   1.5 secs

```

```

Output pane
Data Output Explain Messages History
QUERY PLAN
text
23 Hash Join (cost=61222.90..123508.22 rows=48593 width=7) (actual time=418.484..776.680 rows=53584 loops=1)
  Buffers: shared hit=26619
24   --> Hash (cost=57620.30..57620.30 rows=207248 width=14) (actual time=274.108..274.108 rows=207329 loops=1)
  Output: db2 accident information.accident index
  Buckets: 131072 Batches: 1 Memory Usage: 3355kB
  Buffers: shared hit=4068, temp written=44
  --> Seq Scan on public.db2 accident information db2 accident information 1 (cost=0.0..57620.30 rows=207248 width=14) (actual time=0.546..231.597 rows=207329 loops=1)
  Output: db2 accident information 1.accident index
  Filter: ((db2 accident information 1.date >= '2010-01-01'::date) AND (db2 accident information 1.date <= '2012-12-31'::date) AND ((db2 accident information 1.first road class)::text = 'A'::text))
  Rows Removed by Filter: 1709945
  Buffers: shared hit=24068
33   --> Hash Join (cost=61222.90..123508.22 rows=48593 width=7) (actual time=418.484..776.680 rows=53584 loops=1)
34   Output: db2 vehicle information.make
35   Hash Cond: (db2 vehicle information.accident index)::text = (db2 accident information.accident index)::text
36   Buffers: shared hit=50687, temp read=2346 written=2340
37   --> Seq Scan on public.db2 vehicle information (cost=0.0..53531.06 rows=449448 width=21) (actual time=0.004..300.171 rows=450531 loops=1)
38   Output: db2 vehicle information.make, db2 vehicle information.accident index
39   Filter: ((db2 vehicle information.l.age band of driver)::text = '26 - 35'::text)
40   Rows Removed by Filter: 1726674
41   Buffers: shared hit=26619
42   --> Hash (cost=57620.30..57620.30 rows=207248 width=14) (actual time=273.397..273.397 rows=207329 loops=1)
43   Output: db2 accident information.accident index
44   Buckets: 131072 Batches: 1 Memory Usage: 3355kB
45   Buffers: shared hit=4068, temp written=44
46   --> Seq Scan on public.db2 accident information (cost=0.0..57620.30 rows=207248 width=14) (actual time=0.535..230.493 rows=207329 loops=1)
47   Output: db2 accident information.accident index
48   Filter: ((db2 accident information.date >= '2010-01-01'::date) AND (db2 accident information.date <= '2012-12-31'::date) AND ((db2 accident information.first road class)::text = 'A'::text))
49   Rows Removed by Filter: 1709945
50   Buffers: shared hit=24068
51 Planning time: 0.709 ms
52 Execution time: 1574.724 ms
<
OK.                               DOS  In 20. Col 190. Ch 1740      52 rows.   1.5 secs

```

planning time:0.709ms

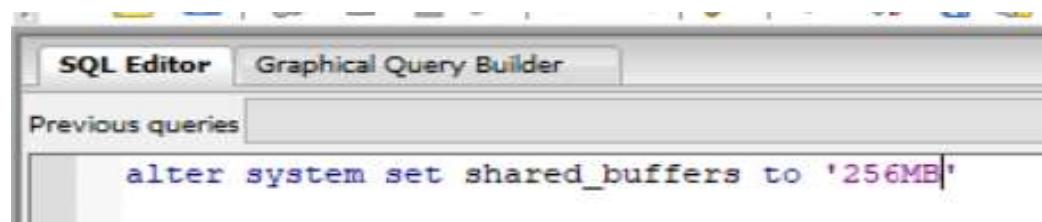
Execution time: 1574.724ms

time:1.5secs

(b) Ρυθμίστε την PostgreSQL ώστε να χρησιμοποιεί ως buffer περισσότερη μνήμη από τη μνήμη RAM του υπολογιστή σας (ικανή ώστε να χωράει όσο γίνεται περισσότερο από το dataset, όλο αν είναι δυνατόν). Έπειτα, εκτελέστε πάλι τις παραπάνω επερωτήσεις και εξηγήστε τι παρατηρείτε. TIP: shared_buffers (π.χ. ALTER SYSTEM SET shared_buffers TO '256MB'; -- απαιτείται επανεκκίνηση του PostgreSQL server).

Λύση:

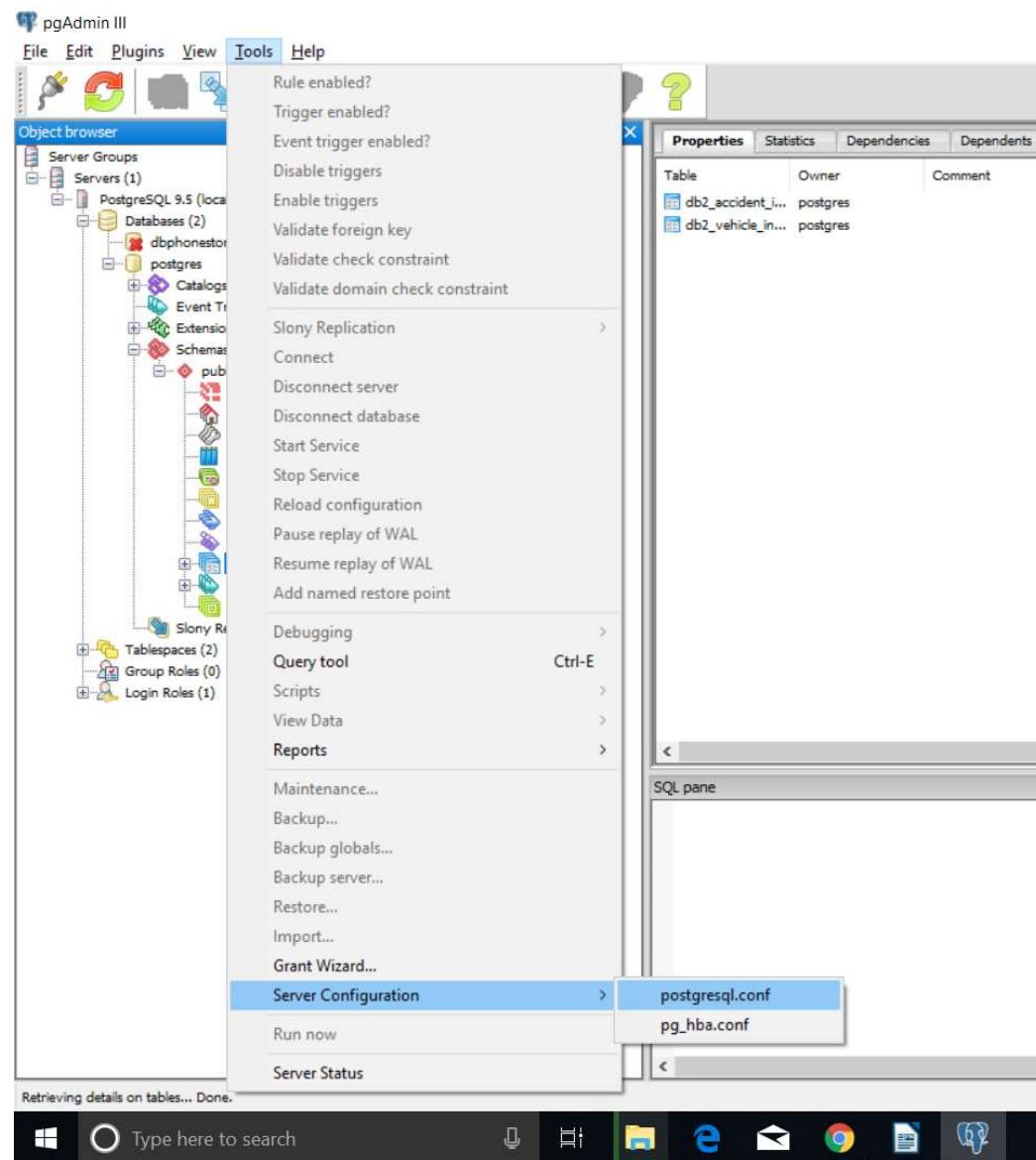
Αυτήν την ρύθμιση γίνεται με δυο τρόπους ο πρώτος τρόπος είναι να ανοίξουμε ένα query και να πληκτρολογήσουμε την εντολή.



```
SQL Editor Graphical Query Builder
Previous queries
alter system set shared_buffers to '256MB'
```

Alter system set shared_buffers to '256MB'

Ο δεύτερος τρόπος είναι να επιλέξουμε εμείς tools→ Server Configuration→ postgresql.conf



Backend Configuration Editor - postgresql.conf on PostgreSQL 9.5 (localhost:5432)

File Edit Help

Setting name	Value	Current value	Comment
geq_threshold	12	12	
constraint_enforcement	partition	partition	on, off, or partition
cursor_tuple_fraction	0.5	0.5	range 0.1-1.0
deadlock_timeout	100	100	range 1-20000
fromCollapse_limit	8	8	1 disables collapsing of explicit
joinCollapse_limit	8	8	
deadlock_timeout	1s	1000	min 10
max_connections	144	64	min 10
max_prepared_transactions	64	64	min 10
autovacuum_work_mem	-1	-1	min 1MB, or -1 to use maintenance_work_mem
dynamic_shared_memory	windows	windows	the default is the first option
huge_pages	try	try	on, off, or try
maintenance_work_mem	64MB	65536	min 1MB
max_prepared_transactions	0	zero	disables the feature
max_stack_depth	2MB	2048	min 200k
shared_buffers	128MB	2097152	min 128kB
temp_buffers	8MB	1024	min 800kB
track_activity_query_size	1024	1024	(change requires restart)
work_mem	4MB	4096	min 64kB
bgwriter_delay	200ms	200	10-10000ms between rounds
bgwriter_lru_maxpages	100	100	0-1000 max buffers written/round
bgwriter_lru_multiplier	2.0	2	0-10.0 multiplier on buffers scanned/round
max_files_per_process	1000	1000	min 25
client_min_messages	notice	notice	values in order of decreasing detail
log_min_duration_statement	<1	<1	<1 is disabled, 0 log all statements
log_min_error_statement	error	error	values in order of decreasing detail
log_min_messages	warning	warning	values in order of decreasing detail
event_source	PostgreSQL	PostgreSQL	
log_destination	stderr	stderr	Valid values are combinations of
log_directory	pg_log	pg_log	directory where log files are written
log_file_mode	0600	0600	creation mode for log files
log_filename	postgre-%Y-%m-%d	postgre-%Y-%m-%d	log file name pattern
log_rotation_age	1d	1440	Automatic rotation of logfiles will
log_rotation_size	10MB	10240	Automatic rotation of logfiles will
log_truncate_on_rotation	off	off	If on, an existing log file is truncated
log_directory_size	on	on	Enable capturing of stderr and csvlog
svrrep_availability	LOCAL0	none	
svrrep_idlet	postgres	postgres	
stats_temp_directory	pg_stat_tmp	pg_stat_tmp	
track_activities	on	on	
track_counts	on	on	
track_functions	none	none, pl, all	

Configuration read from localhost

Θέτω το shared_buffers 16GB όσο είναι και η μνήμη ram μου

Configuration setting "shared_buffers"

Enabled

Value

Comment

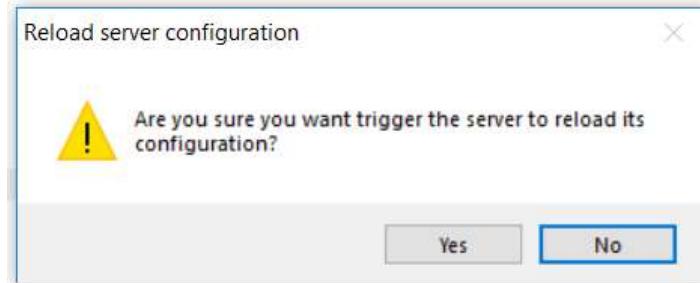
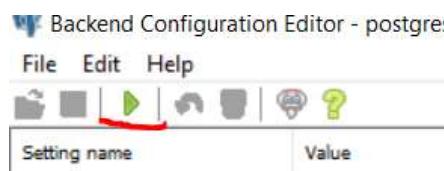
Setting name	Value	Current value	Comment
maintenance_work_mem	64MB	65536	min 1MB
max_prepared_transactions	0	0	zero disables the feature
max_stack_depth	2MB	2048	min 100kB
shared_buffers	16GB	2097152	min 128kB
temp_buffers	8MB	1024	min 800kB
track_activity_query_size	1024	1024	(change requires restart)
work_mem	4MB	4096	min 64kB
bgwriter_delay	200ms	200	10-10000ms between rounds
bgwriter_lru_maxpages	100	100	0-1000 max buffers written/round

Help OK Cancel

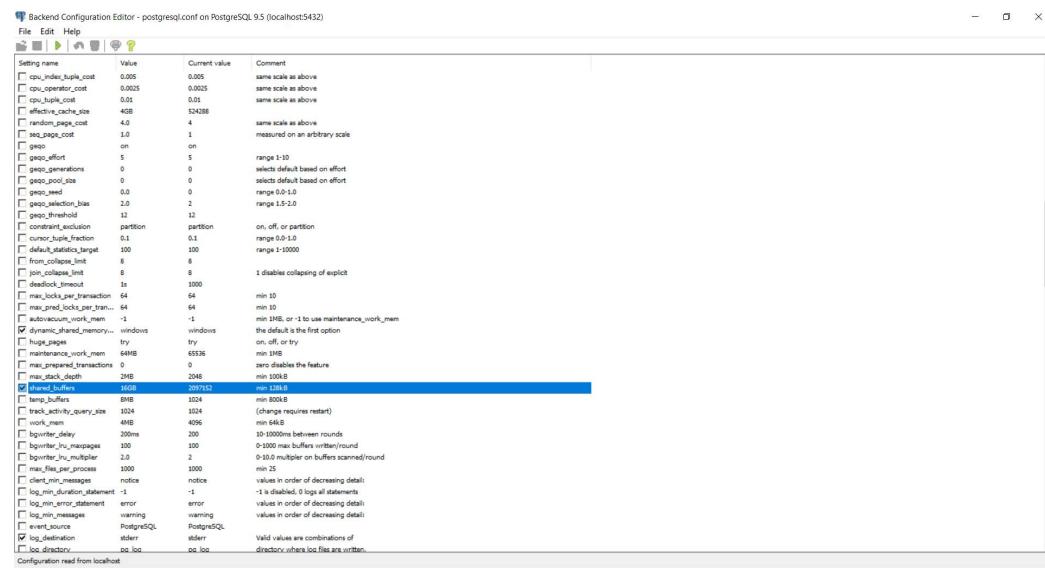
Συστήματα Διαχείρισης Βάσεων Δεδομένων

Και το ρυθμίζουμε στα 16 GB λίγο παραπάνω από την μνήμη ram.

Στην συνέχεια πατάμε να επιβεβαιώσουμε τις αλλαγές.



Και κάνουμε restart στον Postgres sql server



Στην συνέχεια αφού έχω κάνει restart τον server θα ξανά τρέξω τα sql queries μου

Σε κάθε εκτέλεση κάθε ερωτήματος έχω μια **1η εκτέλεση** (αν και λέω πρώτη εκτέλεση μπορεί να έχω τρέξει περισσότερες από τρείς φορές την επερώτηση) από την εκφώνηση:

(πάντα θα εκτελείτε την επερώτηση του λάχιστον δύο φορές και θα κρατάτε τον τελευταίο χρόνο – ο λόγος που δεν κρατάμε απλά τον χρόνο εκτέλεσης της πρώτης φοράς είναι ότι δεν είναι αντιπροσωπευτικός γιατί τα buffers δεν έχουν προλάβει να αρχικοποιηθούν.

i. Βρείτε πόσα ατυχήματα έγιναν ανά κατηγορία δρόμου (road class).

1η Εκτέλεση(αυτή που μας ενδιαφέρει και κρατάμε(αν και λέω πρώτη εκτέλεση μπορεί να έχω τρέξει περισσότερες από τρείς φορές την επερώτηση))

The screenshot shows the pgAdmin 4 interface with the following details:

- SQL Editor:** Contains the query:


```
select first_road_class, count(accident_index) as Accidents_by_road_class
from db2_accident_information
group by first_road_class |
```
- Output pane:** Shows the execution plan and results.
 - Planning time: 0.172 ms
 - Execution time: 490.710 ms
 - Result table (estimated):

first_road_class	Accidents_by_road_class
1	1000
2	1000
3	1000
4	1000
5	1000
6	1000
7	1000
8	1000
9	1000
10	1000
11	1000
12	1000
13	1000
14	1000
15	1000
16	1000
17	1000
18	1000
19	1000
20	1000
21	1000
22	1000
23	1000
24	1000
25	1000
26	1000
27	1000
28	1000
29	1000
30	1000
31	1000
32	1000
33	1000
34	1000
35	1000
36	1000
37	1000
38	1000
39	1000
40	1000
41	1000
42	1000
43	1000
44	1000
45	1000
46	1000
47	1000
48	1000
49	1000
50	1000
51	1000
52	1000
53	1000
54	1000
55	1000
56	1000
57	1000
58	1000
59	1000
60	1000
61	1000
62	1000
63	1000
64	1000
65	1000
66	1000
67	1000
68	1000
69	1000
70	1000
71	1000
72	1000
73	1000
74	1000
75	1000
76	1000
77	1000
78	1000
79	1000
80	1000
81	1000
82	1000
83	1000
84	1000
85	1000
86	1000
87	1000
88	1000
89	1000
90	1000
91	1000
92	1000
93	1000
94	1000
95	1000
96	1000
97	1000
98	1000
99	1000
100	1000

Output pane

Data Output		Explain	Messages	History
QUERY PLAN				
text				
1	HashAggregate (cost=52827.11..52827.17 rows=6 width=19) (actual time=490.627..490.628 rows=6 loops=1)			
2	Output: first road class, count(accident index)			
3	Group Key: db2 accident information.first road class			
4	Buffers: shared hit=24068			
5	-> Seq Scan on public.db2 accident information (cost=0.00..43240.74 rows=1917274 width=19) (actual time=0.009..99.565 rows=1917274 loops=1)			
6	Output: empty, accident index, first road class, accident severity, date, urban or rural area, weather conditions, year, inscotland			
7	Buffers: shared hit=24068			
8	Planning time: 0.172 ms			
9	Execution time: 490.710 ms			

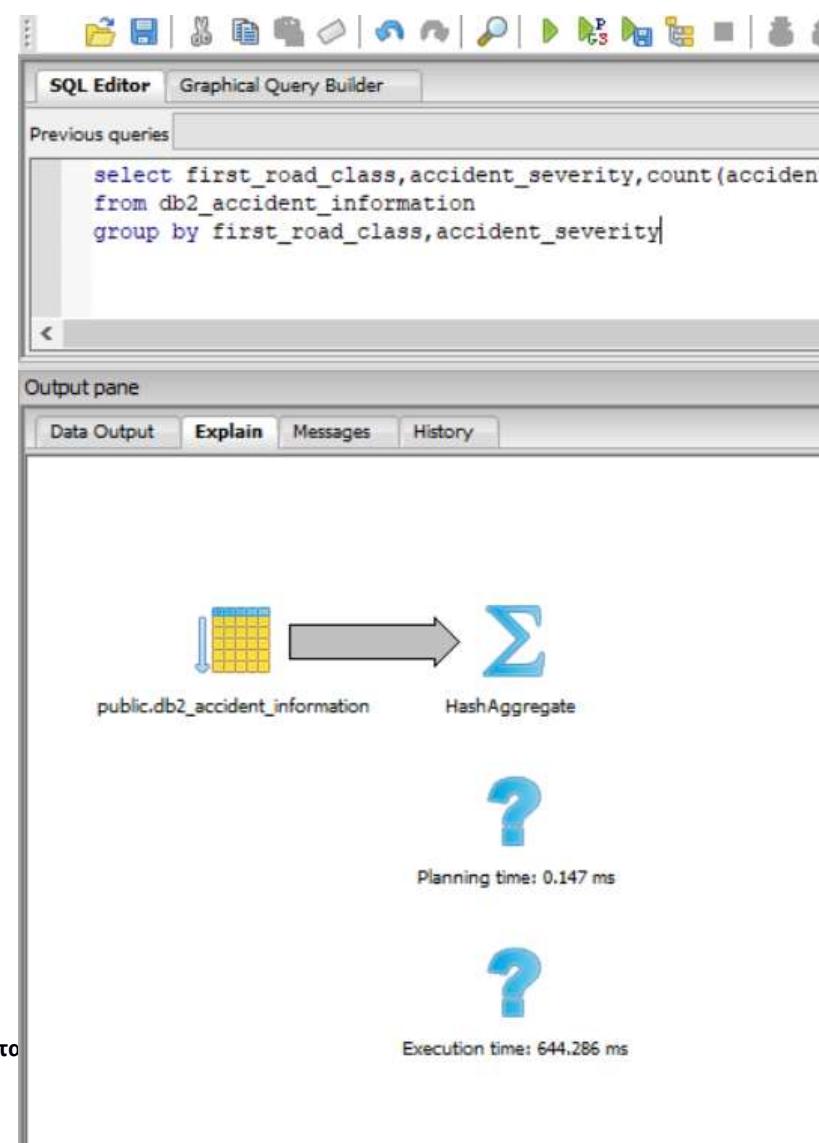
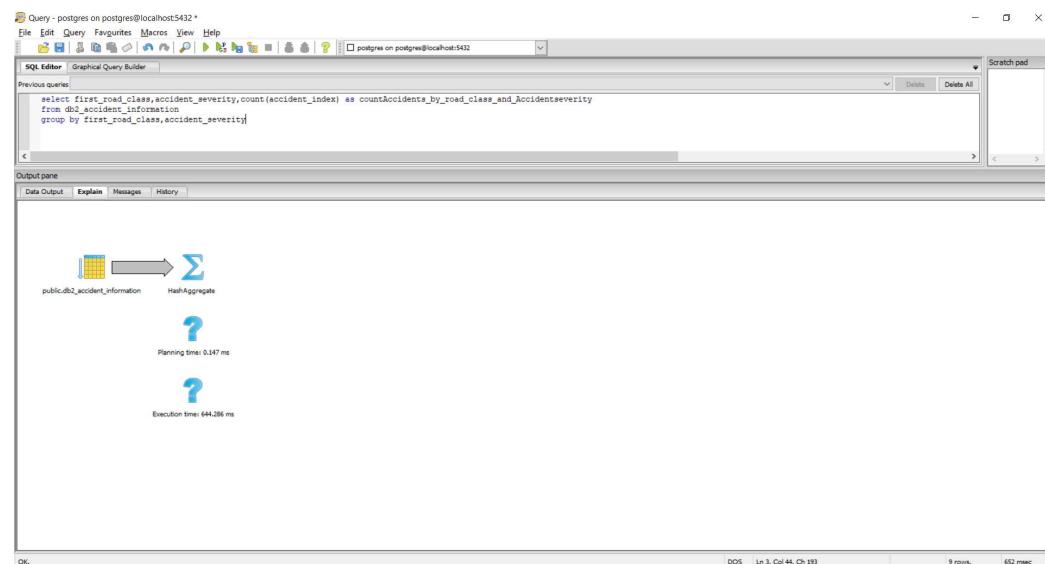
Planning time:0.172ms

Execution time:490.710ms

time:516ms

ii. Πόσα είναι τα ατυχήματα ανά κατηγορία δρόμου και ανά κατηγορία ατυχήματος (Accident Severity)

1η Εκτέλεση(αυτή που μας ενδιαφέρει και κρατάμε(αν και λέω πρώτη εκτέλεση μπορεί να έχω τρέξει περισσότερες από τρείς φορές την επερώτηση))



```

Output pane
Data Output Explain Messages History
QUERY PLAN
text
1 HashAggregate (cost=57620.30..57620.49 rows=18 width=26) (actual time=644.106..644.206 rows=18 loops=1)
  Output: first road class, accident severity, count(accident index)
  Group Key: db2 accident information.first road class, db2 accident information.accident severity
  Buffers: shared hit=24068
  -> Seq Scan on public.db2 accident information (cost=0.00..43240.74 rows=1917274 width=26) (actual time=0.009..103.448 rows=1917274 loops=1)
    Output: empty, accident index, first road class, accident severity, date, urban or rural area, weather conditions, year, inscotland
    Buffers: shared hit=24068
Planning time: 0.147 ms
Execution time: 644.286 ms

```

Planning time:0.147ms

Execution time:644.286ms

time:652ms

iii. Βρείτε πόσα ατυχήματα έγιναν σε αστική περιοχή πριν από την 1/1/2010 και η ηλικιακή κατηγορία του οδηγού είναι 26-35.

1η Εκτέλεση(αυτή που μας ενδιαφέρει και κρατάμε(αν και λέω πρώτη εκτέλεση μπορεί να έχω τρέξει περισσότερες από τρείς φορές την επερώτηση))

Query - postgres@localhost:5432*

File Edit Query Favourites Macros View Help

SQL Editor Graphical Query Builder

Scratch pad

Previous queries:

```
select count(db2_accident_information.accident_index) as UrbanAccidents_lens2010_1_1_agebetween2635
from db2_accident_information
inner join db2_Vehicle_Information
on db2_accident_information.accident_index=db2_Vehicle_Information.accident_index
where db2_accident_information.urban_or_rural_area='Urban' and db2_accident_information.date<'2010/1/1' and db2_Vehicle_Information.age_band_of_driver='26 - 35'
```

Output pane

Data Output Explain Messages History

public.db2_vehicle_information

public.db2_accident_information

Hash Join

Aggregate

Planning time: 0.993 ms

Execution time: 1147.315 ms

OK

DOS Ln 2, Col 12, Ch 113 23 rows. 1.1 sec

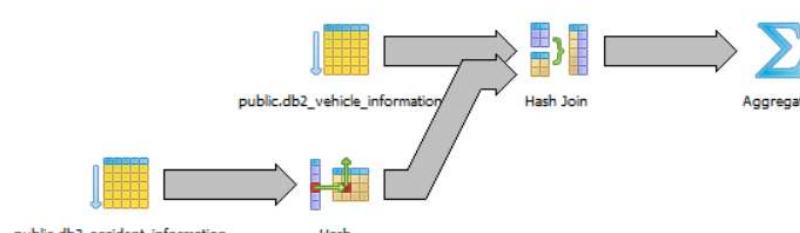
SQL Editor | Graphical Query Builder

Previous queries

```
select count(db2_accident_information.accident_index) as UrbanAccidents_less2010_l_l_agebetween2
from db2_accident_information
inner join db2_Vehicle_Information
on db2_accident_information.accident_index=db2_Vehicle_Information.accident_index
where db2_accident_information.urban_or_rural_area='Urban' and db2_accident_information.date<'2010-12-31'
```

Output pane

Data Output | Explain | Messages | History



The diagram illustrates the query execution plan. It starts with two tables: 'public.db2_vehicle_information' and 'public.db2_accident_information'. The 'public.db2_vehicle_information' table is processed through a 'Hash' operation, which generates a hash key for each row. This hashed data is then joined with the 'public.db2_accident_information' table via a 'Hash Join' operation. Finally, the result of the join is aggregated using an 'Aggregate' operation, represented by a large Greek letter Σ.

Planning time: 0.993 ms

Execution time: 1147.315 ms

```

Output pane
Data Output Explain Messages History
QUERY PLAN
text
1 Aggregate (cost=127243.84..127243.85 rows=1 width=14) (actual time=1146.778..1146.778 rows=1 loops=1)
  2   Output: count(db2 accident information.accident index)
  3   Buffers: shared hit=50687, temp read=4031 written=4001
  4   -> Hash Join (cost=62835.40..126906.45 rows=134956 width=14) (actual time=471.035..1141.118 rows=86298 loops=1)
  5     Output: db2 accident information.accident index
  6     Hash Cond: ((db2 vehicle information.accident index)::text = (db2 accident information.accident index)::text)
  7     Buffers: shared hit=50687, temp read=4031 written=4001
  8     -> Seq Scan on public.db2 vehicle information (cost=0.00..53834.06 rows=449448 width=14) (actual time=0.010..300.605 rows=450531 loops=1)
  9       Output: db2 vehicle information.accident index
 10      Filter: (db2 vehicle information.age band of driver)::text = '26 - 35'::text)
 11      Rows Removed by Filter: 1726674
 12      Buffers: shared hit=26619
 13     -> Hash (cost=52827.11..52827.11 rows=575703 width=14) (actual time=442.720..442.720 rows=573888 loops=1)
 14       Output: db2 accident information.accident index
 15       Buckets: 131072 Batches: 16 Memory Usage: 2645KB
 16       Buffers: shared hit=24068, temp written=2233
 17     -> Seq Scan on public.db2 accident information (cost=0.00..52827.11 rows=575703 width=14) (actual time=0.008..298.486 rows=573888 loops=1)
 18       Output: db2 accident information.accident index
 19       Filter: ((db2 accident information.date < '2010-01-01'::date) AND ((db2 accident information.urban or rural area)::text = 'Urban'::text))
 20       Rows Removed by Filter: 1343386
 21       Buffers: shared hit=24068
 22 Planning time: 0.993 ms
 23 Execution time: 1147.315 ms

```

Planning time:0.993ms

Execution time:1147.315ms

time:1.1ms

iv. Βρείτε πόσα ατυχήματα έγιναν σε αγροτική περιοχή το έτος 2012 και η ηλικιακή κατηγορία του οδηγού είναι 36-45.

1η Εκτέλεση(αυτή που μας ενδιαφέρει και κρατάμε(αν και λέω πρώτη εκτέλεση μπορεί να έχω τρέξει περισσότερες από τρείς φορές την επερώτηση))

```

Query - postgres on localhost:5432+
File Edit Query Favorites Macros View Help
SQL Editor Graphical Query Builder
Scratch pad
Output pane
Data Output Explain Messages History
Planning time: 1.013 ms
Execution time: 587.034 ms

```

Query - postgres on postgres@localhost:5432 *

File Edit Query Favourites Macros View Help

SQL Editor Graphical Query Builder

Previous queries

```
SELECT * FROM public.db2_accident_information
  AS RuralAccidents_Year2012_agebetween3645
  WHERE accident_index = db2_Vehicle_Information.accident_index
    AND accident_information.urban_or_rural_area = 'Rural'
    AND accident_information.date > '2012/1/1' AND db2_Vehicle_Information.vehicle_type = 'Car'
```

Output pane

Data Output Explain Messages History

Planning time: 1.013 ms

Execution time: 587.034 ms

```

Output pane
Data Output Explain Messages History
QUERY PLAN
text
1  Count(*). (cost=11365.41 rows=113509.42 width=14) (actual time=586.410..586.410 rows=1 loops=1)
2    Output: count(db2 accident information.accident index)
3    Buffers: shared hit=45067
4      -> Hash Join (cost=55276.46..113879.32 rows=12036 width=14) (actual time=426.119..585.604 rows=12535 loops=1)
5        Output: db2 accident information.accident index
6        Hash Cond: ((db2 vehicle information.accident index)::text = (db2 accident information.accident index)::text)
7        Buffers: shared hit=50687
8          -> Seq Scan on public.db2 vehicle information (cost=0.00..53834.06 rows=439578 width=14) (actual time=0.013..268.943 rows=435686 loops=1)
9            Output: db2 vehicle information.empty, db2 vehicle information.accident index, db2 vehicle information.age band of driver, db2 vehicle information.make, db2
10           vehicle information.empty, db2 vehicle information.accident index, db2 vehicle information.age band of vehicle, db2 vehicle information.make, db2
11           vehicle information.empty, db2 vehicle information.accident index, db2 vehicle information.age band of driver)::text
12           Rows Removed by Filter: 1741519
13           Buffers: shared hit=26619
14          -> Hash (cost=57620.30..57620.30 rows=52495 width=14) (actual time=244.498..244.498 rows=50066 loops=1)
15            Output: db2 accident information.accident index
16            Buckets: 65536 Batches: 1 Memory Usage: 2762kB
17            Buffers: shared hit=24068
18              -> Seq Scan on public.db2 accident information (cost=0.00..57620.30 rows=52495 width=14) (actual time=146.941..235.217 rows=50066 loops=1)
19                Output: db2 accident information.accident index
20                Filter: ((db2 accident information.date > '2012-01-01'::date) AND ((db2 accident information.date < '2012-12-31'::date) AND ((db2 accident information.urban or rural area)::text = 'Rural'::text))
21                Rows Removed by Filter: 1867208
22                Buffers: shared hit=24068
23 Planning time: 1.013 ms
23 Execution time: 587.034 ms

```

DOS Ln 5. Col 208 Ch 456 23 rows. 592 msec

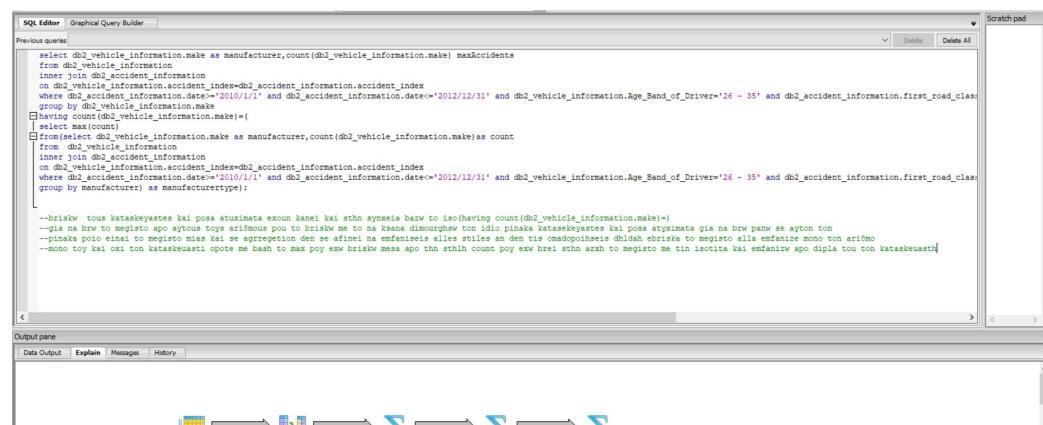
Planning time:1.013ms

Execution time:587.034ms

time:592msec

v. Βρείτε τον κατασκευαστή του οχήματος που έχει τα περισσότερα ατυχήματα μεταξύ των ετών 2010 και 2012, η ηλικιακή κατηγορία των οδηγών είναι 26- 35 και η κατηγορία δρόμου είναι A.

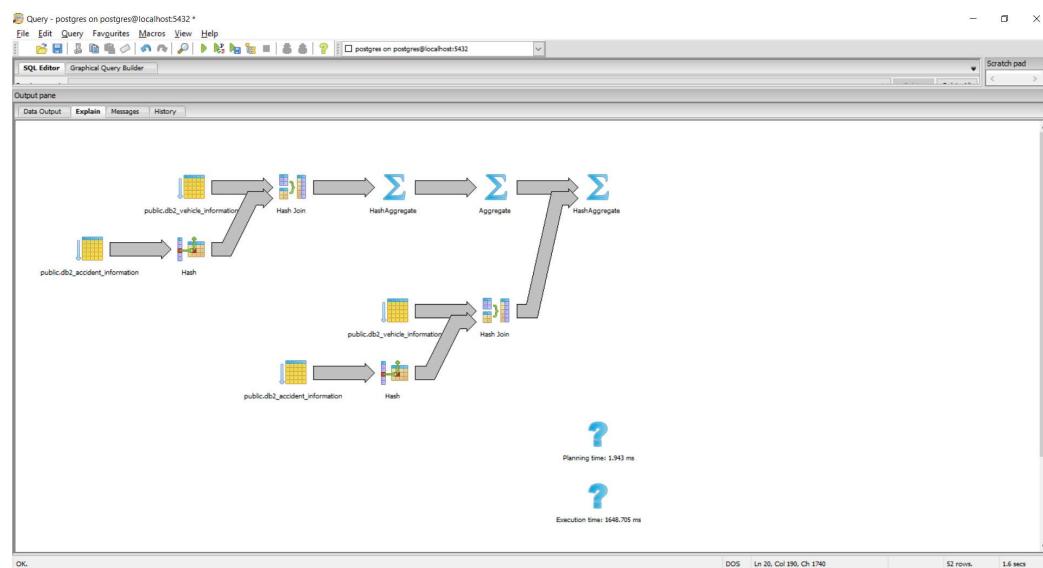
1η Εκτέλεση(αυτή που μας ενδιαφέρει και κρατάμε(αν και λέω πρώτη εκτέλεση μπορεί να έχω τρέξει περισσότερες από τρείς φορές την επερώτηση))



```

SQL Editor | Graphical Query Builder
Previous queries
select db2_vehicle_information.make as manufacturer, count(db2_vehicle_information.make) maxAccidents
from db2_vehicle_information
inner join db2_accident_information
on db2_vehicle_information.accident_index=db2_accident_information.accident_index
where db2_accident_information.date>='2010/1/1' and db2_accident_information.date<='2012/12/31' and db2_vehicle_information.Age_Band_of_Driver='26 - 35' and db2_accident_information.first_road_class='A'
group by db2_vehicle_information.make
having count(db2_vehicle_information.make)=(
| select max(count)
| from (select db2_vehicle_information.make as manufacturer, count(db2_vehicle_information.make)as count
from db2_vehicle_information
inner join db2_accident_information
on db2_vehicle_information.accident_index=db2_accident_information.accident_index
where db2_accident_information.date>='2010/1/1' and db2_accident_information.date<='2012/12/31' and db2_vehicle_information.Age_Band_of_Driver='26 - 35' and db2_accident_information.first_road_class='A'
group by manufacturer) as manufacturerstype;
--briskw tous kataskeuyates kai posa arximata exoun kamei kai sthn synexia hase to iso(having count(db2_vehicle_information.make))
--gia na hrw to megisto apo ayton toyto arximous pou to briskw me to na kanei dimoungwv ton idio pinaka kataskeuyantes kai posa arxymata gia na hrw panw se ayton ton
--pinaka poto einai to megisto mias kai se aggregation den se afinei na emfanizeis allies stiles an den tis omdopoiheis dhlith ekriska to megisto alla emfanize mono ton arxismo
--mono toy toxi kai to kataskeuasti opote me hash to max pou exw briskw mera apo thn sthli count pou exw hrei sthn arxh to megisto me tin isotita kai emfanizi apo diplo tou ton kataskeuasti

```



Planning time:1.943ms

Execution time:1648.705



Planning time: 1.943 ms



Execution time: 1648.705 ms

```

Output pane
Data Output Explain Messages History
QUERY PLAN
1 HashAggregate  (cost=16724.86 .. 247654.48 rows=50 width=21) (actual time=1647.403..1647.407 rows=1 loops=1)
  Output: db2 vehicle information.make, count(db2 vehicle information.make)
  Group Key: db2 vehicle information.make
  Filter: (count(db2 vehicle information.make) = 20)
  Rows Removed by Filter: 216
  Buffers: shared hit=101374, temp read=4692 written=4680
  InitPlan (for update)
    1-> Aggregate  (cost=123753.26 .. 123752.27 rows=1 width=0) (actual time=799.000..799.001 rows=1 loops=1)
      Output: max(count(db2 vehicle information.l.make))
      Buffers: shared hit=5067, temp read=2346 written=2340
    11-> HashAggregate  (cost=123751.13 .. 123751.63 rows=50 width=7) (actual time=798.953..798.983 rows=217 loops=1)
      Output: db2 vehicle information.l.make, count(db2 vehicle information.l.make)
      Group Key: db2 vehicle information.l.make
      Filter: shared hit=5067, temp read=2346 written=2340
      Buffers: shared hit=5067, temp read=2346 written=2340
      Hash Cond: db2 vehicle information.l.accident_index::text = (db2 accident information.l.accident_index)::text
      Buffers: shared hit=5067, temp read=2346 written=2340
      Output: db2 vehicle information.l.accident_index
      Filter: (db2 vehicle information.l.accident_index::text = '26 - 35'::text)
      Rows Removed by Filter: 172674
      Buffers: shared hit=5067, temp read=2346 written=2340
      Hash  (cost=1620.30 .. 16720.30 rows=207248 width=14) (actual time=277.305..277.305 rows=207328 loops=1)
      Output: db2 accident information.l.accident_index
      Buckets: 131072 Batches: 4 Memory Usage: 3355kB
      Buffers: shared hit=24068, temp written=444
      >- Seq Scan on public.db2 vehicle information db2 vehicle information l  (cost=0.00..53834.06 rows=449448 width=21) (actual time=0.009..303.848 rows=50531 loops=1)
      Output: db2 vehicle information.l.make, db2 vehicle information.l.accident_index
      Filter: (db2 vehicle information.l.accident_index::text = '26 - 35'::text)
      Rows Removed by Filter: 172674
      Buffers: shared hit=24068, temp read=2346 written=2340
      Hash  (cost=1620.30 .. 16720.30 rows=207248 width=14) (actual time=277.305..277.305 rows=207329 loops=1)
      Output: db2 accident information.l.accident_index
      Buckets: 131072 Batches: 4 Memory Usage: 3355kB
      Buffers: shared hit=24068, temp written=444
      >- Seq Scan on public.db2 accident information db2 accident information l  (cost=0.00..57620.30 rows=207248 width=14) (actual time=85.258..232.969 rows=207329 loops=1)
      Output: db2 accident information.l.accident_index
      Filter: (db2 accident information.l.date >='2010-01-01'::date) AND (db2 accident information.l.date <='2012-12-31'::date) AND ((db2 accident information.l.first_road_class
      Rows Removed by Filter: 170945
      Buffers: shared hit=24068
<
```

```

SQL Editor Graphical Query Builder
Scratch pad
Previous queries
Output pane
Data Output Explain Messages History
QUERY PLAN
text
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
Planning time: 1.943ms
Execution time: 1648.705
time: 1.6secs
OK
DOS Ln 20, Col 190, Ch 1760
52 rows. 1.6 secs

```

Planning time:1.943ms

Execution time:1648.705

time:1.6secs

Καταγράφουμε τώρα τα στοιχεία σε πίνακες

1ο Πίνακας(όλα είναι by default δεν έχουμε πειράξει τίποτα)

Query	Planning time	Execution time	Total time
i	0.061ms	449.906ms	469msecs
ii	0.147ms	620.265ms	641msecs
iii	0.977ms	1061.916ms	1secs
iv	0.983ms	557.107ms	578msecs
v	0.709ms	1574.724ms	1.5secs

Shared buffers:16GB

Query	Planning time	Execution time	Total time

i	0.172ms	490.710ms	516msecs
ii	0.147ms	644.286ms	652msecs
iii	0.993ms	1147.315ms	1.1secs
iv	1.013ms	587.034ms	592msecs
v	1.943ms	1648.705ms	1.6secs

Δυο πινακες μαζί

Query	Planning time	Planning time share buffers 16GB	Execution time	Execution time share buffers 16GB	Total time	Total time share buffers 16GB
i	0.061ms	0.172ms	449.906ms	490.710ms	469msecs	516msecs
ii	0.147ms	0.147ms	620.265ms	644.286ms	641msecs	652msecs
iii	0.977ms	0.993ms	1061.916ms	1147.315ms	1secs	1.1secs
iv	0.983ms	1.013ms	557.107ms	587.034ms	578msecs	592msecs
v	0.709ms	1.943ms	1574.724ms	1648.705ms	1.5secs	1.6secs

Όπως παρατηρούμε επειδή έχουμε αυξήσει το μέγεθος της μνήμης τόσο ώστε να είναι μεγαλύτερο από την μνήμη ram στα 16GB παρατηρούμε μια αύξηση του χρόνου εκτέλεσης των επερωτήσεων και αυτό είναι φυσιολογικό αφού δεν κάνει όλους τους υπολογισμούς μέσα στην ram.

Στην συνέχεια για όλες τις υπόλοιπες εκτελέσεις (μπορεί να έχω τρέξει περισσότερες από τρεις φορές τις επερωτήσεις κάθε φορά απλά δεν το γράφω κάθε φορά που τρέχω ένα query))

(c) Ρυθμίστε την PostgreSQL έτσι ώστε να χρησιμοποιεί όλη την επεξεργαστική ισχύ του υπολογιστή σας. Έπειτα, εκτελέστε πάλι τις παραπάνω επερωτήσεις και εξηγήστε τι παρατηρείτε. TIP: max_parallel_workers_per_gather

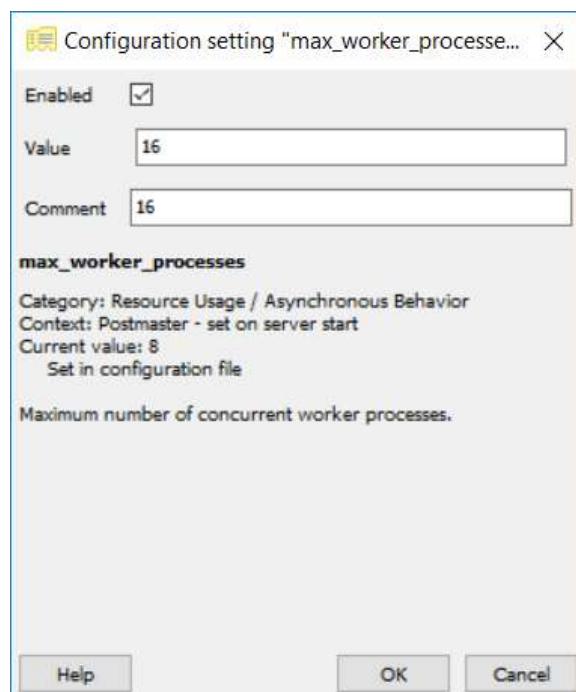
Παω εκει που λέει tools→Server Configuration → postgresql.conf

Backend Configuration Editor - postgresql.conf on PostgreSQL 9.5 (localhost:5432)

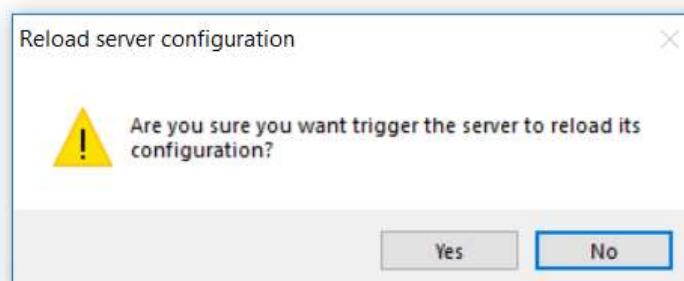
Setting name	Value	Current value	Comment
track_io_timing	off	off	
log_executor_stats	off	off	
log_executor_all	off	off	
log_executor_min	off	off	
log_statement_stats	off	off	
log_statement	off	off	
log_statement_min	0	0	range 0-30000, in microseconds
commit_delay	5	5	range 1-1000
sync_page_delay	on	on	turns forced synchronization on or off
full_page_writes	on	on	recovery from partial page writes
synchronous_commit	on	on	synchronization level
wal_buffers	-1	2048	min 32B; -1 sets based on shared_buffers
wal_compression	off	off	enable compression of full-page writes
wal_level	minimal	minimal	minimal, archive, hot_standby, or logical
wal_log_hints	off	off	also log write hints of non-critical updates
wal_log_method	fsync	open_datasync	the default is the first option
wal_writer_delay	200ms	200	1-10000 milliseconds
transform_null_equals	off	off	
allow_system_table_mods	off	off	
ignore_checksum_failure	off	off	
ignore_index_defects	off	off	
post_auth_delay	0	0	
pre_auth_delay	0	0	
trace_notify	off	off	
trace_recovery_messages	log	log	
trace_stat	off	off	
zero_damaged_pages	off	off	
config_file	C:/Program Files...	C:/Program Files...	use data in another directory
data_directory	ConfigDir	C:/Program Files...	use data in another directory
external_pid_file	ConfigDir/pg_hb...	C:/Program Files...	host-based authentication file
hba_file	ConfigDir/pg_id...	C:/Program Files...	ident configuration file
ident_file	ConfigDir/pg_id...	C:/Program Files...	use data in another directory
effective_io_concurrency	0	0	1-1000; 0 disables prefetching
max_worker_processes	8	8	8
archive_command	(disabled)		
archive_mode	off	off	enables archiving: off, on, or always
archive_timeout	0	0	force a logfile segment switch after this
vacuum_cost_delay	0	0	0-100 milliseconds
vacuum_cost_limit	200	200	1-10000 credits
vacuum_cost_page_dirty	20	20	0-10000 credits
vacuum_cost_page_hlt	1	1	0-10000 credits
vacuum_cost_page_min	10	10	0-10000 credits
temp_file_size	-1	-1	1mb per session temp file space

Configuration read from localhost

trace_notify	off		
trace_recovery_messages	log		
trace_sort	off		
zero_damaged_pages	off		
config_file	C:/Program Files...		
data_directory	ConfigDir	C:/Program Files...	use data in another directory
external_pid_file			
hba_file	ConfigDir/pg_hb...	C:/Program Files...	host-based authentication file
ident_file	ConfigDir/pg_id...	C:/Program Files...	ident configuration file
effective_io_concurrency	0	0	1-1000; 0 disables prefetching
max_worker_processes	8	8	8
archive_command	(disabled)		
archive_mode	off	off	enables archiving: off, on, or always
archive_timeout	0	0	force a logfile segment switch after this
vacuum_cost_delay	0	0	0-100 milliseconds
vacuum_cost_limit	200	200	1-10000 credits
vacuum_cost_page_dirty	20	20	0-10000 credits



Θέτουμε τους maxworkers 16



<input type="checkbox"/> <code>data_directory</code>	ConfigDir	C:/Program Files...	use data in another directory
<input type="checkbox"/> <code>external_pid_file</code>			
<input type="checkbox"/> <code>hba_file</code>	ConfigDir/pg_hb...	C:/Program Files...	host-based authentication file
<input type="checkbox"/> <code>ident_file</code>	ConfigDir/pg_id...	C:/Program Files...	ident configuration file
<input type="checkbox"/> <code>effective_io_concurrency</code>	0	0	1-1000; 0 disables prefetching
<input checked="" type="checkbox"/> <code>max_worker_processes</code>	16	8	16
<input type="checkbox"/> <code>archive_command</code>	(disabled)		
<input type="checkbox"/> <code>archive_mode</code>	off	off	enables archiving; off, on, or always
<input type="checkbox"/> <code>archive_timeout</code>	0	0	force a logfile segment switch after this
<input type="checkbox"/> <code>vacuum_cost_delay</code>	0	0	0-100 milliseconds

i. Βρείτε πόσα ατυχήματα έγιναν ανά κατηγορία δρόμου (road class).

The screenshot shows the pgAdmin 4 interface with a query editor window. The query is:

```
select first_road_class, count(accident_index) as Accidents_by_road_class
from db2_accident_information
group by first_road_class;
```

The output pane displays the execution plan and results:

- Planning time: 0.149 ms
- Execution time: 469.450 ms
- 9 rows.
- 485 msec

Query - postgres on postgres@localhost:5432 *

File Edit Query Favourites Macros View Help

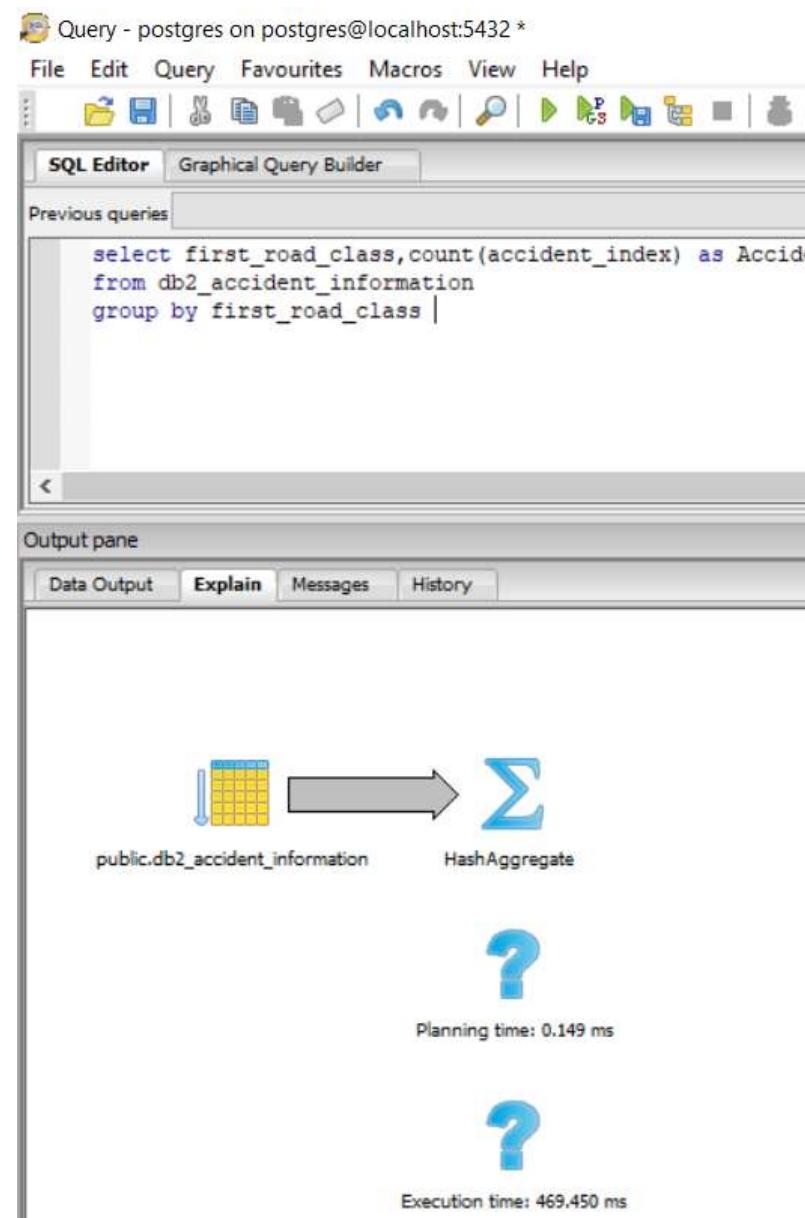
SQL Editor Graphical Query Builder

Previous queries

```
select first_road_class, count(accident_index) as Accide
from db2_accident_information
group by first_road_class |
```

Output pane

Data Output Explain Messages History



The screenshot shows a PostgreSQL graphical query builder interface. In the SQL Editor tab, a query is written to select accident counts by road class from a table named db2_accident_information. Below the editor, the Output pane displays the execution plan. The plan consists of two main stages: a HashAggregate operation on the public.db2_accident_information table, followed by a planning step and an execution step. The HashAggregate stage is represented by a grid icon and a summation symbol, indicating it groups data and performs aggregation. The planning step shows a question mark icon with a planning time of 0.149 ms. The execution step shows a question mark icon with an execution time of 469.450 ms.

```

Output pane
Data Output Explain Messages History
QUERY PLAN
text
1 HashAggregate (cost=52827.11..52827.17 rows=6 width=19) (actual time=469.357..469.358 rows=6 loops=1)
  2   Output: first_road_class, count(accident_index)
  3   Group Key: db2_accident_information.first_road_class
  4   Buffers: shared hit=24068
  5     -> Seq Scan on public.db2_accident_information (cost=0.00..43240.74 rows=1917274 width=19) (actual time=0.009..95.593 rows=1917274 loops=1)
  6       Output: empty, accident_index, first_road_class, accident_severity, date, urban or rural area, weather_conditions, year, inscotland
  7       Buffers: shared hit=24068
  8 Planning time: 0.149 ms
  9 Execution time: 469.450 ms
  
```

Planning time 0.149ms

Execution time 469.450ms

time 485msecs

ii. Πόσα είναι τα ατυχήματα ανά κατηγορία δρόμου και ανά κατηγορία ατυχήματος (Accident Severity)

```

File Edit Query Favorites Macros View Help
File Edit Query Favorites Macros View Help
SQL Editor Graphical Query Builder
Scratch pad
Previous queries
select first_road_class,accident_severity,count(accident_index) as countAccidents_by_road_class_and_AccidentSeverity
from db2_accident_information
group by first_road_class,accident_severity;
  
```

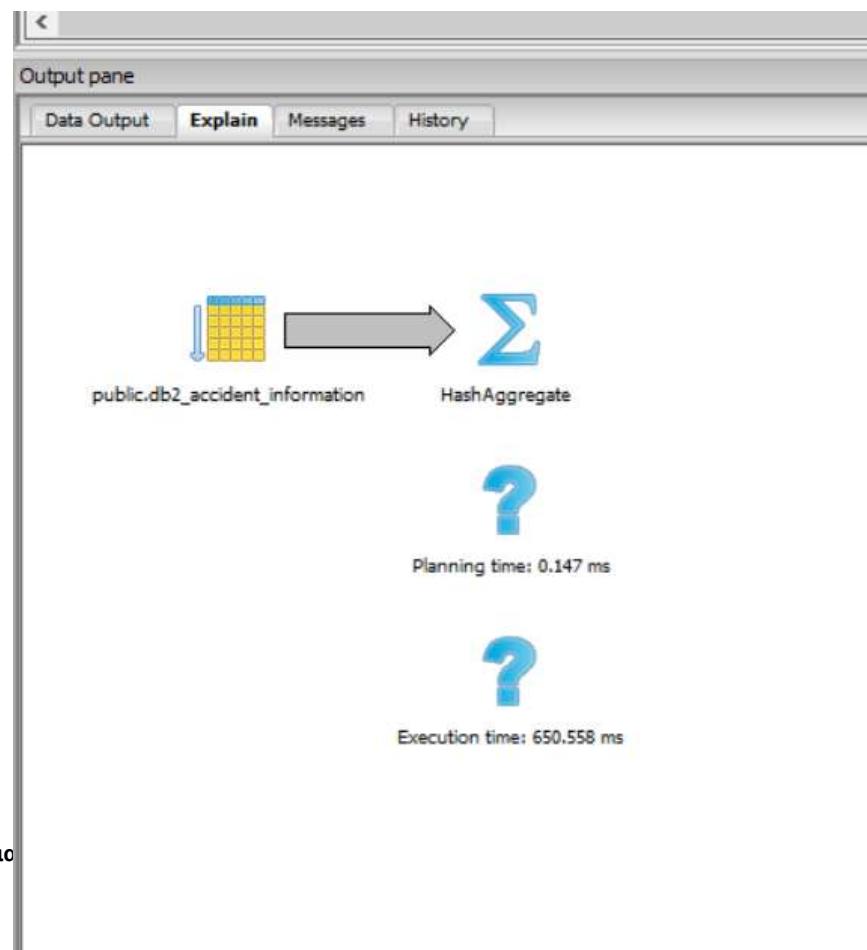
Output pane

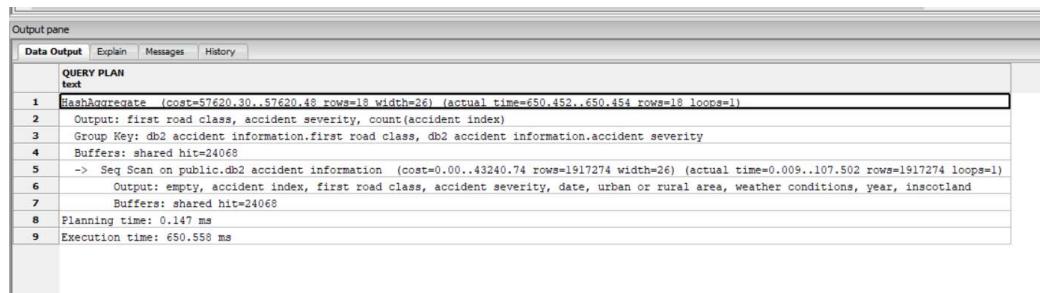
Data Output Explain Messages History

public.db2_accident_information HashAggregate

Planning time: 0.147 ms

Execution time: 650.558 ms





The screenshot shows a software interface for managing database queries. At the top, there's a menu bar with tabs like 'Data Output', 'Explain', 'Messages', and 'History'. Below the menu is a section titled 'QUERY PLAN' with the 'text' option selected. The main area displays a numbered list of SQL execution steps:

```
1 HashAggregate (cost=57620.30..57620.48 rows=18 width=26) (actual time=650.452..650.454 rows=18 loops=1)
  2   Output: first road class, accident severity, count(accident index)
  3   Group Key: db2 accident information.first road class, db2 accident information.accident severity
  4   Buffers: shared hit=24068
  5   -> Seq Scan on public.db2 accident information (cost=0.00..43240.74 rows=1917274 width=26) (actual time=0.009..107.502 rows=1917274 loops=1)
  6     Output: empty, accident index, first road class, accident severity, date, urban or rural area, weather conditions, year, inscotland
  7     Buffers: shared hit=24068
  8 Planning time: 0.147 ms
  9 Execution time: 650.558 ms
```

Planning time 0.147ms

Execution time 450.558ms

time 671msecs

iii. Βρείτε πόσα ατυχήματα έγιναν σε αστική περιοχή πριν από την 1/1/2010 και η ηλικιακή κατηγορία του οδηγού είναι 26-35.

Query - postgres@localhost:5432*

```
File Edit Query Favourites Macros View Help
SQL Editor Graphical Query Builder
Scratch pad
Previous queries:
select count(db2_accident_information.accident_index) as UrbanAccidents_lens2010_1_1_agebetween2635
from db2_Vehicle_Information
inner join db2_Accident_Information
on db2_Accident_Information.accident_index=db2_Vehicle_Information.accident_index
where db2_Accident_Information.urban_or_rural_area='Urban' and db2_Accident_Information.date='2010/1/1' and db2_Vehicle_Information.age_band_of_driver='26 - 35'!
```

Output pane

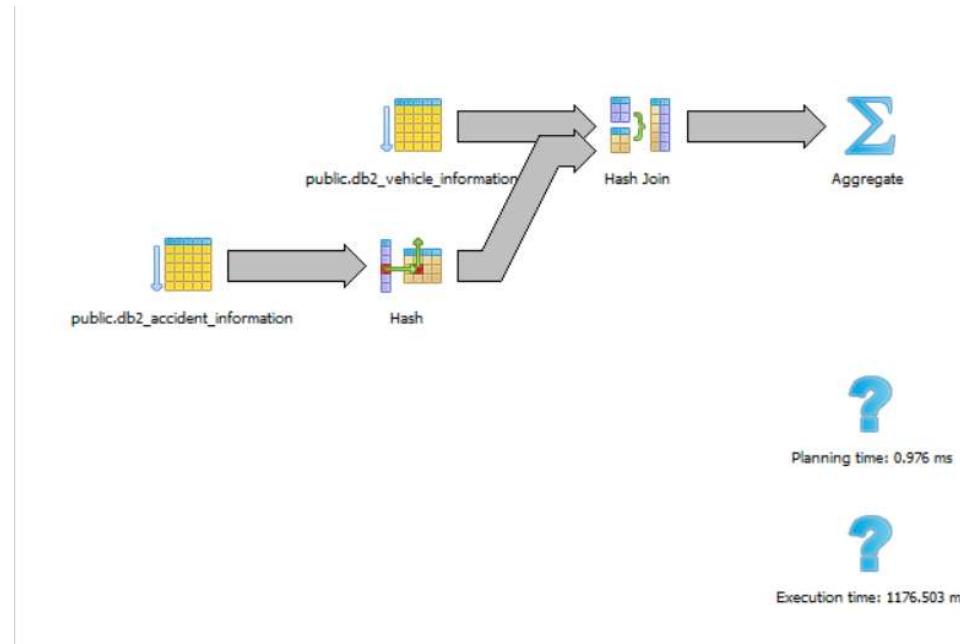
Data Output Explain Messages History

Planning time: 0.976 ms

Execution time: 1176.503 ms

OK

DOS Ln 5 Col 161 Ch 412 23 rows. 1.1 sec



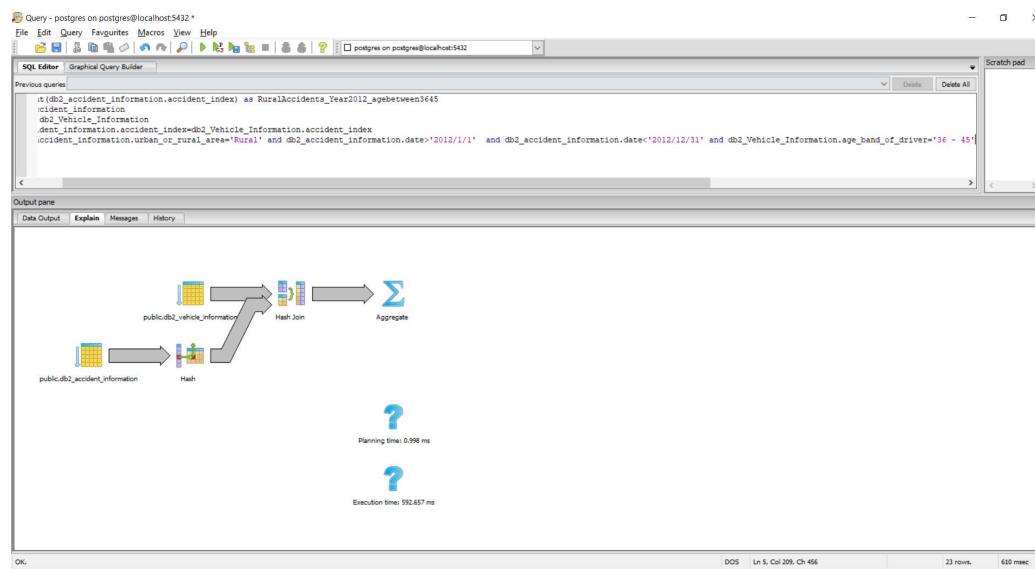
Output pane	
	Data Output Explain Messages History
QUERY PLAN	
1	text
1	Aggregate (cost=127243.84..127243.85 rows=1 width=14) (actual time=1175.932..1175.932 rows=1 loops=1)
2	Output: count(db2 accident information.accident index)
3	Buffers: shared hit=50687, temp read=4031 written=4001
4	-> Hash Join (cost=62835.40..126906.45 rows=134956 width=14) (actual time=458.808..1169.526 rows=86298 loops=1)
5	Output: db2 accident information.accident index
6	Hash Cond: ((db2 vehicle information.accident index)::text = (db2 accident information.accident index)::text)
7	Buffers: shared hit=50687, temp read=4031 written=4001
8	-> Seq Scan on public.db2 vehicle information (cost=0.00..53834.06 rows=449448 width=14) (actual time=0.010..297.010 rows=450531 loops=1)
9	Output: db2 vehicle information.accident index
10	Filter: ((db2 vehicle information.age band of driver)::text = '26 - 35'::text)
11	Rows Removed by Filter: 1726674
12	Buffers: shared hit=26619
13	-> Hash (cost=52827.11..52827.11 rows=575703 width=14) (actual time=430.537..430.537 rows=573888 loops=1)
14	Output: db2 accident information.accident index
15	Buckets: 131072 Batches: 16 Memory Usage: 2645kB
16	Buffers: shared hit=24068, temp written=2223
17	-> Seq Scan on public.db2 accident information (cost=0.00..52827.11 rows=575703 width=14) (actual time=0.008..288.810 rows=573888 loops=1)
18	Output: db2 accident information.accident index
19	Filter: ((db2 accident information.date < '2010-01-01'::date) AND ((db2 accident information.urban or rural area)::text = 'Urban'::text))
20	Rows Removed by Filter: 1343386
21	Buffers: shared hit=24068
22	Planning time: 0.976 ms
23	Execution time: 1176.503 ms

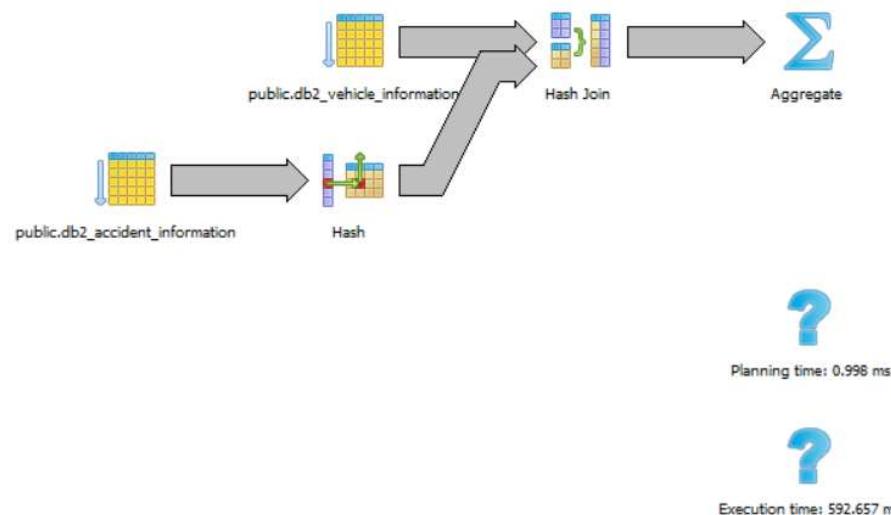
Planning time 0.976ms

Execution time 1176.503ms

time 1.1secs

iv. Βρείτε πόσα ατυχήματα έγιναν σε αγροτική περιοχή το έτος 2012 και η ηλικιακή κατηγορία του οδηγού είναι 36-45.





Planning time 0.998ms

Execution time 592.657ms

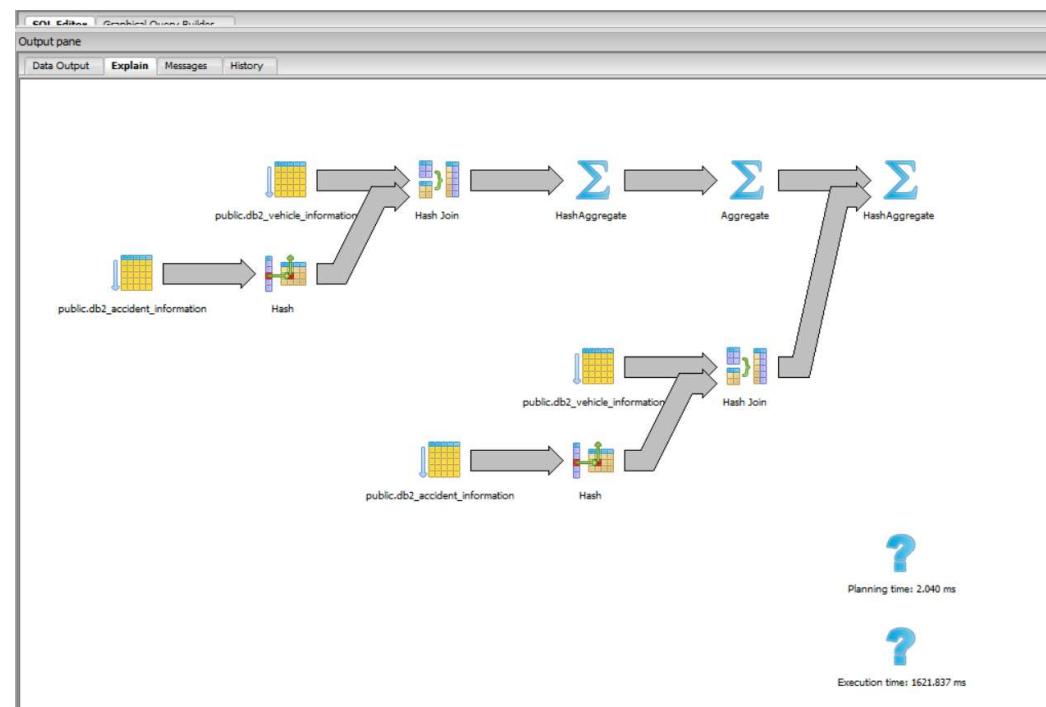
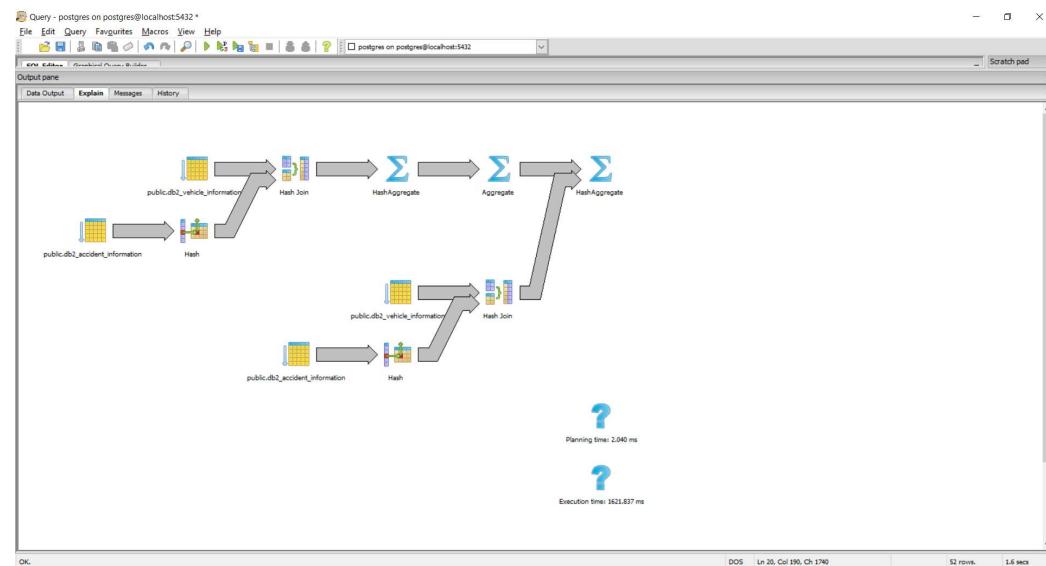
time 610msec

```

Output pane
Data Output Explain Messages History
QUERY PLAN
1  Seq Scan on public.db2_vehicle_information (cost=0.00..53834.04 rows=113895 width=14) (actual time=592.650..592.650 rows=1 loops=1)
2    Output: count(db2 accident information.accident index)
3    Buffers: shared hit=50687
4  ->  Hash Join  (cost=55276.48..113879.32 rows=12036 width=14) (actual time=433.753..591.229 rows=12535 loops=1)
5    Output: db2 accident information.accident index
6    Hash Cond: (db2 vehicle information.accident index)::text = (db2 accident information.accident index)::text
7    Buffers: shared hit=26615
8      ->  Seq Scan on public.db2 vehicle information (cost=0.00..53834.04 rows=439578 width=14) (actual time=0.014..267.886 rows=435686 loops=1)
9        Output: db2 vehicle information.empty, db2 vehicle information.accident index, db2 vehicle information.age band of driver, db2 vehicle information.age of vehicle, db2 vehicle information.make, db2
10       Filter: ((db2 vehicle information.age band of driver)::text = '34 - 45'::text)
11       Rows Removed by Filter: 1741519
12       Buffers: shared hit=26615
13      ->  Hash  (cost=57620.30..57620.30 rows=52495 width=14) (actual time=253.456..253.456 rows=50066 loops=1)
14        Output: db2 accident information.accident index
15        Buckets: 65536  Batches: 1  Memory Usage: 2762KB
16        Buffers: shared hit=24068
17        ->  Seq Scan on public.db2 accident information (cost=0.00..57620.30 rows=52495 width=14) (actual time=157.005..244.403 rows=50066 loops=1)
18          Output: db2 accident information.accident index
19          Filter: ((db2 accident information.date > '2012-01-01'::date) AND ((db2 accident information.date < '2012-12-31'::date) AND ((db2 accident information.urban or rural area)::text = 'Rural'::text))
20          Rows Removed by Filter: 1867208
21          Buffers: shared hit=24068
22 Planning time: 0.998 ms
23 Execution time: 592.657 ms

```

v. Βρείτε τον κατασκευαστή του οχήματος που έχει τα περισσότερα ατυχήματα μεταξύ των ετών 2010 και 2012, η ηλικιακή κατηγορία των οδηγών είναι 26-35 και η κατηγορία δρόμου είναι A.



Query - postgres@localhost:5432 *

File Edit Query Favorites Macros View Help

Output pane

Data Output Explain Messages History

QUERY PLAN

```

20      text
        Output: db2.vehicle_information.l.make, db2.vehicle_information.l.accident_index
        Filter: ((db2.vehicle_information.l.age band of driver)::text = '26 - 35':text)
        Rows Removed by Filter: 1724674
        Buffers: shared hit=2619
21      -
22      -
23      -
24      -> Hash (cost=57620.30..57620.30 rows=207248 width=14) (actual time=204.517..204.517 rows=207329 loops=1)
        Output: db2.accident_information.l.accident_index
        Buckets: 131072 Batches: 4 Memory Usage: 3355kB
        Buffers: shared hit=24068, temp written=644
25      -
26      -
27      -
28      -> Seq Scan on public.db2.accident_information.l (cost=0.00..57620.30 rows=207248 width=14) (actual time=85.242..239.813 rows=207329 loops=1)
        Output: db2.accident_information.l.accident_index
        Filter: ((db2.accident_information.l.date >='2010-01-01':date) AND (db2.accident_information.l.date <='2012-12-31':date) AND ((db2.accident_information.l.first_road_class
        Rows Removed by Filter: 1709945
        Buffers: shared hit=24068
29      -
30      -
31      -
32      -
33      -> Hash Join (cost=61222.90..123505.22 rows=45583 width=7) (actual time=439.382..805.001 rows=53584 loops=1)
        Output: db2.vehicle_information.make
34      -
35      Hash Cond: ((db2.vehicle_information.accident_index)::text = (db2.accident_information.accident_index)::text)
        Buffers: shared hit=50687, temp read=2346 written=2340
36      -
37      -> Seq Scan on public.db2.vehicle_information (cost=0.00..53584.06 rows=449448 width=21) (actual time=0.01..307.626 rows=450531 loops=1)
        Output: db2.vehicle_information.make, db2.vehicle_information.accident_index
38      -
39      Filter: ((db2.vehicle_information.age band of driver)::text = '26 - 35':text)
        Rows Removed by Filter: 1724674
        Buffers: shared hit=2619
40      -
41      -
42      -> Hash (cost=57620.30..57620.30 rows=207248 width=14) (actual time=287.412..287.412 rows=207329 loops=1)
        Output: db2.accident_information.accident_index
43      -
44      -
45      -
46      -> Seq Scan on public.db2.accident_information (cost=0.00..57620.30 rows=207248 width=14) (actual time=83.198..241.887 rows=207329 loops=1)
        Output: db2.accident_information.accident_index
47      -
48      Filter: ((db2.accident_information.date >='2010-01-01':date) AND (db2.accident_information.date <='2012-12-31':date) AND ((db2.accident_information.first_road_class)::text = 'A'::text)
        Rows Removed by Filter: 1709945
        Buffers: shared hit=24068
51      Planning time: 2.040 ms
52      Execution time: 1621.837 ms

```

Planning time 2.040ms

Execution time 1621.837ms

time 1.6sec

Max parallel workers=16

Query	Planning time	Execution time	Total time
i	0.149ms	469.450ms	485msecs
ii	0.147ms	450.558ms	671msecs
iii	0.976ms	1176.503ms	1.1secs
iv	0.998ms	592.657ms	610msecs
v	2.040ms	1621.837ms	1.6secs

Query	Planning time	Planning time share buffers 16GB	Planning time max parallel workers 16	Execution time share buffers 16GB	Execution time share buffers 16GB	Execution time max parallel workers 16	Total time	Total time share buffers 16GB	Total time share buffers 16GB
i	0.061ms	0.172ms	0.149ms	449.906ms	490.710ms	469.450ms	469msecs	516msecs	485msecs
ii	0.147ms	0.147ms	0.147ms	620.265ms	644.286ms	450.558ms	641msecs	652msecs	671msecs
iii	0.977ms	0.993ms	0.976ms	1061.916ms	1147.315ms	1176.503ms	1secs	1.1secs	1.1secs
iv	0.983ms	1.013ms	0.998ms	557.107ms	587.034ms	592.657ms	578msecs	592msecs	610msecs
v	0.709ms	1.943ms	2.040ms	1574.724ms	1648.705ms	1621.837ms	1.5secs	1.6secs	1.6secs

Τα αποτελέσματα που συγκρίνουμε είναι με έντονα μαύρα γράμματα. Αυτό που μπορούμε να παρατηρήσουμε είναι ότι υπάρχει μια βελτίωση χρόνου σε σχέση με το προηγούμενο παράδειγμα που χρησιμοποιούμε για shared buffers =16GB αλλά οι χρόνοι μας δεν είναι τόσο βελτιωμένη όσο όταν δουλεύαμε όταν είμασταν μέσα στην ram. Αυτό είναι φυσιολογικό γιατί έχουμε αυξήσει τους max_parallel_workers οπότε περιμέναμε μια βελτίωση χρόνου αλλά όχι τόσο μεγάλη όσο όσο θα ήταν αν εργαζόμασταν μέσα στη ram.

(d) Δημιουργήστε τα κατάλληλα ευρετήρια στη ΒΔ για να τρέξουν οι παραπάνω επερωτήσεις πιο γρήγορα. Για κάθε ευρετήριο που θα δημιουργήσετε θα εξηγήσετε τους λόγους για τους οποίους επιλέξατε τον συγκεκριμένο τύπο ευρετηρίου και το πώς βοηθάει στην βελτίωση του χρόνου εκτέλεσης.

```

Query - postgres on localhost:5432*
File Edit Query Favorites Macros View Help
SQL Editor Graphical Query Builder
Scratch pad
Previous queries:
create index dateBtree on db2_accident_information using btree(date);
create index urban_or_rural_areashash on db2_accident_information using hash(urban_or_rural_area);
create index age_band_of_driverHash on db2_vehicle_information using hash(age_band_of_driver);
Output pane
Data Output Explain Messages History
WARNING: hash indexes are not WAL-logged and their use is discouraged
WARNING: hash indexes are not WAL-logged and their use is discouraged
Query returned successfully with no result in 14:01 minutes.
OK
DOS Ln 3, Col 97, Ch 271 14:01 minutes
  
```

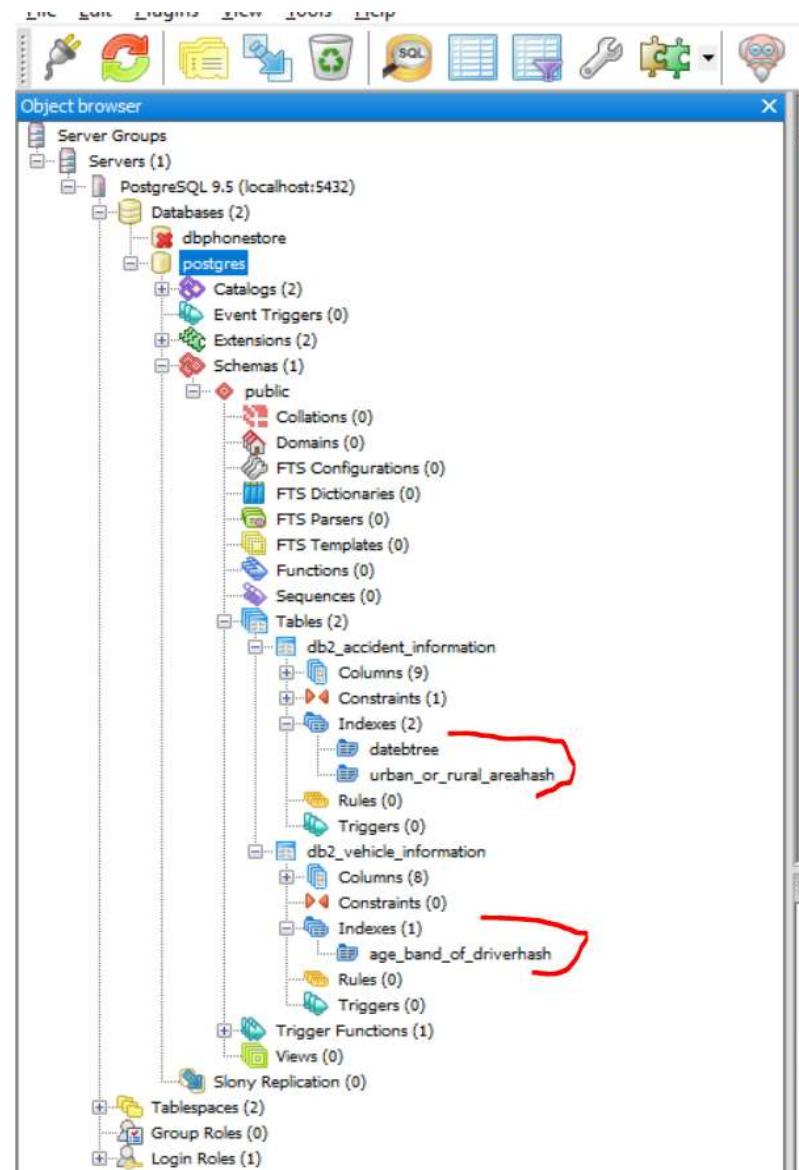
Θα δημιουργήσω ένα btree ευρετήριο για την στήλη **date** από τον πίνακα **db2_accident_information**. Χρησιμοποιώ btrees γιατί διαχειρίζονται καλύτερα ισότητες και οτιδήποτε έχει να κάνει με εύρος. Η postgresql προτιμάει να χρησιμοποιήσει btrees όταν μια στήλη εμπλέκεται σε συγκρίσεις. Δηλαδή όταν βλέπουμε τα παρακάτω σύμβολα.

μικροτερο(<=) ,μικροτερο ισο (<) ,μεγαλυτερο(>) ,μεγαλυτερο ισο(>=) και ισοτητες.

create index dateBtree on db2_accident_information using btree(date);

Στην συνέχεια δημιουργώ hash ευρετήρια για την στήλη **urban_or_rural_area** του πίνακα **db2_accident_information** και για την στήλη **age_band_of_driver** του πίνακα **db2_vehicle_information**.

Ό χρόνος που μας πήρε να φτιάξουμε τα ευρετήρια είναι 14 λεπτά και 01 δευτερόλεπτα. Μας πήρε τοσο χρόνο γιατί τα hash ευρετήρια χρειάζονται αρκετό χρόνο για να δημιουργηθούν.



Τρέχω ξανά τα επερωτήματα.

i. Βρείτε πόσα ατυχήματα έγιναν ανά κατηγορία δρόμου (road class).

The screenshot shows a PostgreSQL graphical query builder interface. In the SQL Editor tab, the following query is displayed:

```
select first_road_class, count(accident_index) as Accidents_by_road_class
from db2.accident_information
group by first_road_class;
```

In the Output pane, the Explain tab shows the execution plan:

```

public.db2.accident_information  HashAggregate
                                 |
                                 ?
                                 Planning time: 0.190 ms
                                 ?
                                 Execution time: 470.544 ms

```

The plan indicates a HashAggregate operation on the public.db2.accident_information table. The planning time is 0.190 ms and the execution time is 470.544 ms.

The screenshot shows the PostgreSQL output pane with the Explain tab selected. The query plan is displayed in text format:

```

QUERY PLAN
text
1 HashAggregate  (cost=52827.11..52827.17 rows=6 width=19) (actual time=470.466..470.467 rows=6 loops=1)
  2   Output: first road class, count(accident index)
  3   Group Key: db2.accident_information.first road class
  4   Buffers: shared hit=24068
  5   ->  Seq Scan on public.db2.accident_information  (cost=0.00..43240.74 rows=1917274 width=19) (actual time=0.009..96.962 rows=1917274 loops=1)
  6     Output: empty, accident index, first road class, accident severity, date, urban or rural area, weather conditions, year, inscotland
  7     Buffers: shared hit=24068
  8   Planning time: 0.190 ms
  9   Execution time: 470.544 ms

```

Planning time:0.190ms

Execution time:470.544ms

Time:488msec

ii. Πόσα είναι τα ατυχήματα ανά κατηγορία δρόμου και ανά κατηγορία ατυχήματος (Accident Severity).

The screenshot shows the pgAdmin III interface with a query editor window open. The query is:

```
select first_road_class,accident_severity,count(accident_index) as countAccidents_by_road_class_and_accidentseverity
from db2_accident_information
group by first_road_class,accident_severity;
```

The results pane displays the execution plan and statistics:

- Planning time: 0.196 ms
- Execution time: 641.895 ms

At the bottom, it shows the command was run on 'DOS' at line 3, column 44, character 193, resulting in 9 rows in 655 msec.

Output pane

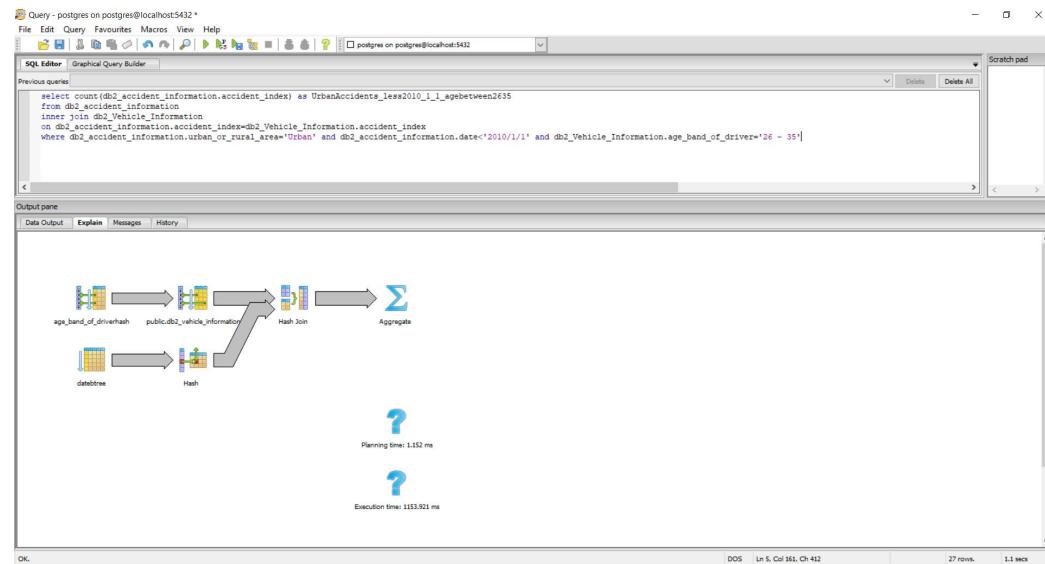
	Data Output	Explain	Messages	History
1	QUERY PLAN			
2	text			
3	1 HashAggregate (cost=57620.30..57620.48 rows=18 width=26) (actual time=641.804..641.807 rows=18 loops=1)			
4	2 Output: first road class, accident severity, count(accident index)			
5	3 Group Key: db2 accident information.first road class, db2 accident information.accident severity			
6	4 Buffers: shared hit=24068			
7	5 -> Seq Scan on public.db2 accident information (cost=0.00..43240.74 rows=1917274 width=26) (actual time=0.009..103.373 rows=1917274 loops=1)			
8	6 Output: empty, accident index, first road class, accident severity, date, urban or rural area, weather conditions, year, inscotland			
9	7 Buffers: shared hit=24068			
	8 Planning time: 0.198 ms			
	9 Execution time: 641.895 ms			

Planning time:0.198ms

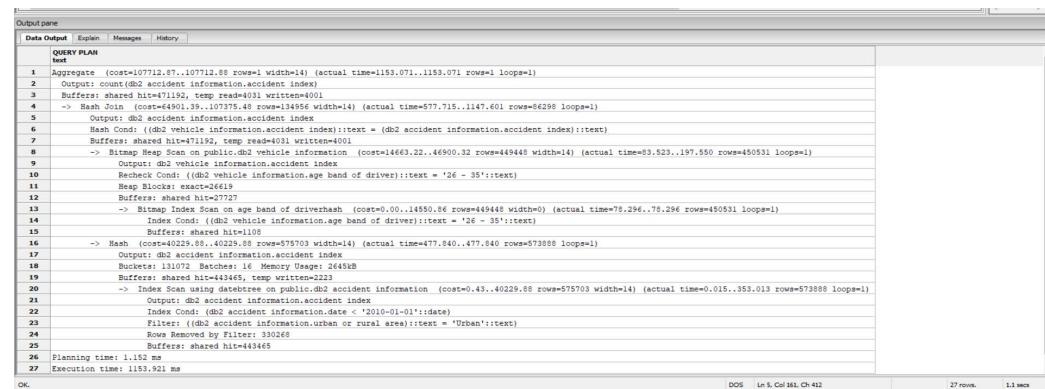
Execution time:641.895ms

time:655msec

iii. Βρείτε πόσα ατυχήματα έγιναν σε αστική περιοχή πριν από την 1/1/2010 και η ηλικιακή κατηγορία του οδηγού είναι 26-35.



Όπως βλέπουμε έχει χρησιμοποιήσει τα ευρετήρια μας **urban_or_rural_areaHash** και **dateBtree**

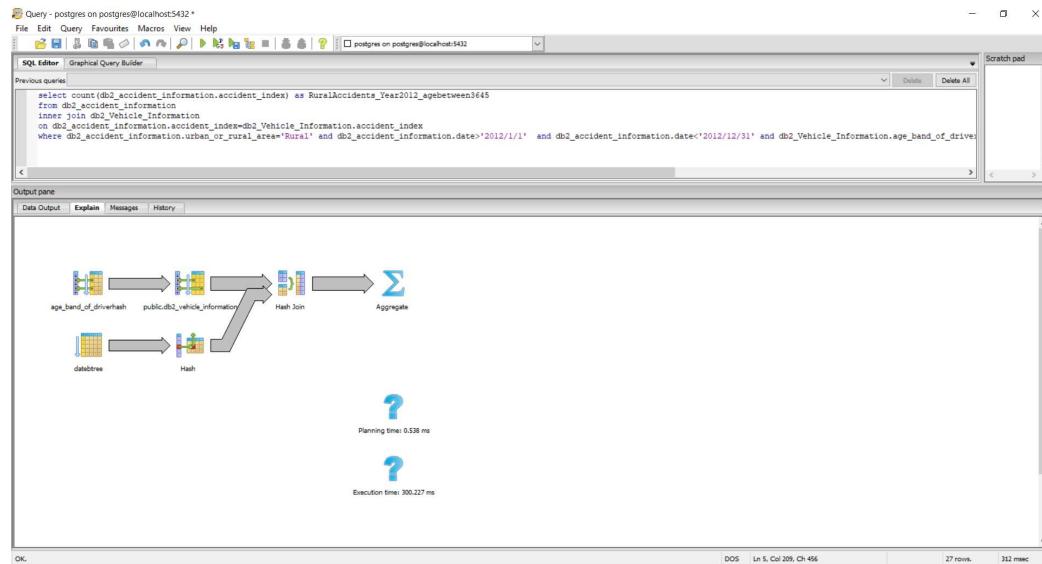


Planning time:1.152ms

Execution time:1153.921ms

time:1.1secs

iv. Βρείτε πόσα ατυχήματα έγιναν σε αγροτική περιοχή το έτος 2012 και η ηλικιακή κατηγορία του οδηγού είναι 36-45.



```

Output pane
Data Output Explain Messages History
QUERY PLAN
Nested loop
1 Aggregate (cost=57029.85..57029.86 rows=1 width=14) (actual time=299.387..299.387 rows=1 loops=1)
   Output: count(db2 accident information.accident index)
3   Buffers: shared hit=99196
4   >- Hash Join (cost=23117.26..56999.76 rows=12036 width=14) (actual time=220.611..299.513 rows=12535 loops=1)
5     Output: db2 accident information.accident index
6     Hash Cond: ((db2 vehicle information.accident index)::text = (db2 accident information.accident index)::text)
7     Buffers: shared hit=99196
8     >- Bitmap Heap Scan on public.db2 vehicle information (cost=14342.73..44456.45 rows=439578 width=14) (actual time=35.638..140.452 rows=435686 loops=1)
9       Output: db2 accident information.accident index
10      Bitmap Heap Scan on public.db2 vehicle information.age band of driver::text = ('36 - 45')::text
11      Hash Cond: ((db2 vehicle information.age band of driver)::text = ('36 - 45')::text)
12      Buffers: shared hit=27491
13      >- Bitmap Index Scan on age band of driverhash (cost=0.00..14232.83 rows=439578 width=0) (actual time=31.166..31.166 rows=435686 loops=1)
14        Index Cond: ((db2 vehicle information.age band of driver)::text = '36 - 45')::text
15        Buffers: shared hit=1072
16        >- Hash (cost=5118.34..5118.34 rows=52495 width=14) (actual time=86.000..86.000 rows=50066 loops=1)
17          Output: db2 accident information.accident index
18          Buckets: 65536 Batches: 1 Memory Usage: 27424B
19          Buffers: shared hit=71505
20          >- Index Scan using datbase on public.db2 accident information (cost=0.43..8118.34 rows=52495 width=14) (actual time=0.019..71.942 rows=50066 loops=1)
21            Index Cond: ((db2 accident information.date > '2012-01-01'::date) AND (db2 accident information.date < '2012-12-31'::date))
22            Output: db2 accident information.accident index
23            Filter: ((db2 accident information.urban or rural area)::text = 'Rural')::text
24            Rows Removed by Filter: 94999
25            Buffers: shared hit=71505
26 Planning time: 0.538 ms
27 Execution time: 300.227 ms

```

OK.

DOS Ln 5. Col 209. Ch 496 27 rows. 312 msec

Planning time 0.538ms

Execution time:300.227ms

time:312msec

ν. Βρείτε τον κατασκευαστή του οχήματος που έχει τα περισσότερα ατυχήματα μεταξύ των ετών 2010 και 2012, η ηλικιακή κατηγορία των οδηγών είναι 26- 35 και η κατηγορία δρόμου είναι A.

Query - postgres@localhost:5432*

```

File Edit Query Favourites Macros View Help
SQL Editor Graphical Query Builder Scratch pad
Previous queries
vehicle_information.make as manufacturer, count(db2_vehicle_information.make) maxAccidents
'while information
db2_accident_information
accident_index
accident_information.date<='2010/11/1' and db2_accident_information.date<='2012/12/31' and db2_vehicle_information.Age_Band_of_Driver='26 - 35' and db2_accident_information.first_road_class='R'
Q_vehicle_information.make=(

) as manufacturer_type;
I open
db2_vehicle_information.make as manufacturer, count(db2_vehicle_information.make)as count
'while information
db2_accident_information
accident_index
accident_information.date<='2010/11/1' and db2_accident_information.date<='2012/12/31' and db2_vehicle_information.Age_Band_of_Driver='26 - 35' and db2_accident_information.first_road_class='R'
manufacturer_type) as manufacturer_type;

κατασκευαστες και ποσα αποτελεσματα έχουν και στην σύνθετη βάση το iso(having count(db2_vehicle_information.make)=

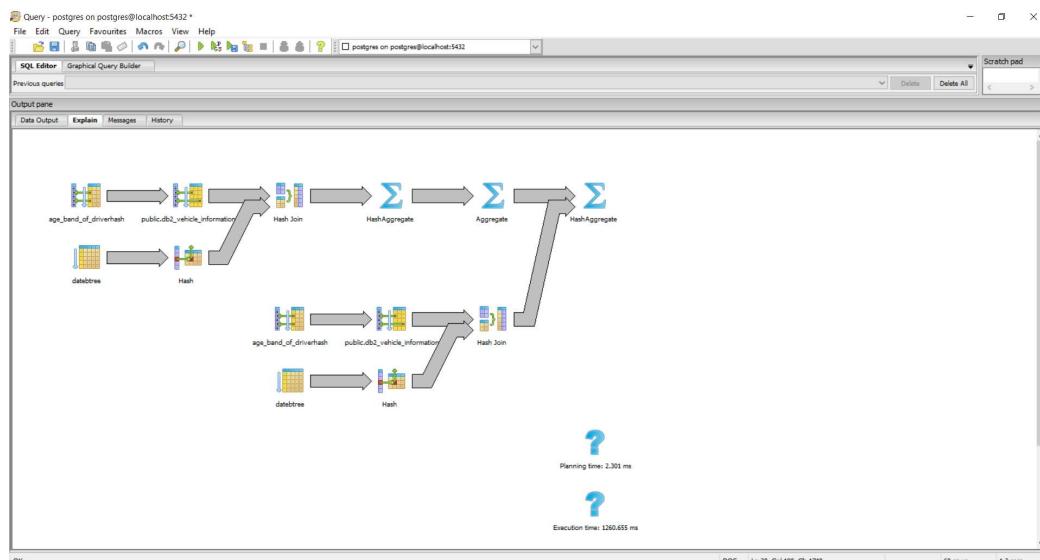
w to megisto apo syntous toy arithmos pou to briwsi me to na krama dimorfizew ton idio pinaka katastasekeyastes kai posa stypimata gia na baw panne se arithmo ton
do einai to megisto mikra kai se apgegretion den se afseis na emfaniseis allies stiles as den tis omedopoleseis dilithi christs to megisto alla emfaniseis me ton arithmo
hai oti ton kataskeusasti opore me baini to max pou to briwsi meseta apo ton within count pou emfani sthn archi to megistou apo tis leitoti kai emfanisi apo dipla tou ton kataskeusasth

```

Output pane

Manufacturer	maxAccidents
FORD	6449

DOS Ln 20, Col 190, Ch 1740 | 1 row. 11 secs



Όπως βλέπουμε χρησιμοποιεί στην επεξεργασία το **datebtree** και **age_band_of_driverHash**

```

Output pane
Data Output Explain Messages History
QUERY PLAN
1 HashAggregate (cost=163395.79..163396.42 rows=50 width=7) (actual time=1259.603..1259.610 rows=1 loops=1)
  Output: db2 vehicle information.make, count(db2 vehicle information.make)
  Group Key: db2 vehicle information.make
  Filter: (count(db2 vehicle information.make) = 50)
  Rows Removed by Filter: 216
  Buffers: shared hit=498234, temp read=4692 written=4680
  InitPlan 1 (returning 50)
  -> Aggregate (cost=61637.73..61637.74 rows=1 width=8) (actual time=605.564..605.564 rows=1 loops=1)
      Output: make, count(db2 vehicle information.l.make)
      Buffers: shared hit=2340, temp read=2340 written=2340
      -> HashAggregate (cost=61634.60..61637.10 rows=50 width=7) (actual time=605.512..605.545 rows=217 loops=1)
          Output: db2 vehicle information.l.make, count(db2 vehicle information.l.make)
          Group Key: db2 vehicle information.l.make
          Buffers: shared hit=249117, temp read=2344 written=2340
          -> Hash Join (cost=40705.33..81393.69 rows=45853 width=7) (actual time=331.190..554.507 rows=53584 loops=1)
              Output: db2 vehicle information.l.make
              Hash Cond: ((db2 vehicle information.l.accident_index)::text = (db2 accident information.l.accident_index)::text)
              Buffers: shared hit=2340, temp read=2340 written=2340
              -> Bitmap Scan on db2 vehicle information.l.accident_index (cost=14663.22..46900.32 rows=449448 width=21) (actual time=34.773..162.138 rows=450531 loops=1)
                  Output: db2 vehicle information.l.accident_index
                  Recheck Cond: (db2 vehicle information.l.accident_index <= '26 - 35':text)
                  Heap Blocks: actual=2619
              Buffers: shared hit=27727
              -> Bitmap Index Scan on age band of driverhash (cost=0.00..14550.86 rows=449448 width=0) (actual time=30.219..30.219 rows=450531 loops=1)
                  Index Cond: (db2 vehicle information.l.age_band_of_driver)::text = '26 - 35':text
                  Buffers: shared hit=1108
              -> Hash (cost=120.00..207325 width=14) (actual time=211.142..211.142 rows=207325 loops=1)
                  Output: db2 accident information.l.accident_index
                  Batches: 131072 Batches: 1 Memory Usage: 335596
                  Buffers: shared hit=22190, temp written=444
                  -> Index Scan using datebookt on db2 accident information.db2 accident information.l (cost=0.43..22439.50 rows=207248 width=14) (actual time=0.024..146.141 rows=207329 loops=1)
                      Output: db2 accident information.l.accident_index
                      Index Cond: ((db2 accident information.l.date > '2010-01-01':date) AND (db2 accident information.l.date <='2012-12-31':date))

```

```

Output pane
Data Output Explain Messages History
QUERY PLAN
text

00      :> OUTPUT: WNG_WC100001110000001:WNC100001110000001 INDEX
01      :Buckets: 131072 Batches: 4 Memory Usage: 3355kB
02      :Buffers: shared hit=221390, temp written=444
03      :-> Index Scan using dateonpublic on public.db2 accident information db2 accident information l (cost=0.43..22439.50 rows=207248 width=14) (actual time=0.024..164.141 rows=207329 loops=1)
04          :    Output: db2 accident information.l.accident_index
05          :    Index Cond: ((db2 accident information.l.date >= '2010-01-01':date) AND (db2 accident information.l.date <= '2012-12-31':date))
06          :    Filter: ((db2 accident information.l.first road class)::text = 'A':text)
07          :    Rows Removed by Filter: 244130
08          :    Buffers: shared hit=221390
09      :-> Hash Join (cost=40705.33..81393.69 rows=45583 width=1) (actual time=379.310..641.963 rows=53584 loops=1)
10          :    Output: db2 vehicle information.name
11          :    Hash Cond: ((db2 vehicle information.accident_index)::text = (db2 accident information.accident_index)::text)
12          :    Buffers: shared hit=249172, temp read=2346 written=2340
13          :-> Bitmap Heap Scan on public.db2 vehicle information (cost=14663.22..46900.32 rows=449448 width=21) (actual time=76.484..202.925 rows=450531 loops=1)
14              :    Output: db2 vehicle information.name, db2 vehicle information.accident_index
15              :    Index Cond: ((db2 vehicle information.age band of drives)::text = '26 - 35':text)
16              :    Heap Block: exact=4661
17              :    Buffers: shared hit=27727
18              :-> Bitmap Index Scan on age band of driverhash (cost=0.00..14550.86 rows=449448 width=0) (actual time=71.486..71.486 rows=450531 loops=1)
19                  :    Index Cond: ((db2 vehicle information.age band of driver)::text = '26 - 35':text)
20                  :    Buffers: shared hit=1108
21          :-> Hash (cost=22439.50..22439.50 rows=207248 width=14) (actual time=217.562..217.562 rows=207329 loops=1)
22              :    Output: db2 accident information.accident_index
23              :    Buckets: 131072 Batches: 4 Memory Usage: 3355kB
24              :    Buffers: shared hit=221390, temp written=444
25              :-> Index Scan using dateonpublic on public.db2 accident information (cost=0.43..22439.50 rows=207248 width=14) (actual time=0.016..169.470 rows=207329 loops=1)
26                  :    Output: db2 accident information.accident_index
27                  :    Index Cond: ((db2 accident information.date >= '2010-01-01':date) AND (db2 accident information.date <= '2012-12-31':date))
28                  :    Filter: ((db2 accident information.first road class)::text = 'A':text)
29                  :    Rows Removed by Filter: 244130
30                  :    Buffers: shared hit=221390
31          :Planning time: 2.301 ms
32          :Execution time: 1260.655 ms


```

Planning time:2.301ms

Execution time:1260.655ms

Time:1.2secs

index times

Query	Planning time	Execution time	Total time
i	0.190ms	470.544ms	488msec
ii	0.198ms	641.895ms	655msec
iii	1.152ms	1153.921ms	1.1sec
iv	0.538ms	300.277ms	312msec
v	2.031ms	1260.655ms	1.2sec

Query	Planning time	Planning time share buffers 16GB	Planning time max parallel workers 16	Planning time index times	Execution time share buffers 16GB	Execution time share buffers 16GB	Execution time max parallel workers 16	Execution time index times	Total time	Total time share buffers 16GB	Total time share buffers 16GB	Total time index times
i	0.061ms	0.172ms	0.149ms	0.190ms	449.906ms	490.710ms	469.450ms	470.544ms	469msec	516msec	485msec	488ms
ii	0.147ms	0.147ms	0.147ms	0.198ms	620.265ms	644.286ms	450.558ms	641.895ms	641msec	652msec	671msec	655ms
iii	0.977ms	0.993ms	0.976ms	1.152ms	1061.916ms	1147.315ms	1176.503ms	1153.921ms	1secs	1.1secs	1.1secs	1.1sec
iv	0.983ms	1.013ms	0.998ms	0.538ms	557.107ms	587.034ms	592.657ms	300.277ms	578msec	592msec	610msec	312ms
v	0.709ms	1.943ms	2.040ms	2.031ms	1574.724ms	1648.705ms	1621.837ms	1260.655ms	1.5secs	1.6secs	1.6secs	1.2sec

Οι μετρήσεις που μας ενδιαφέρουν είναι από το iii, iv, v ερώτημα όπου χρησιμοποιούμε τα ευρετήρια μας και βλέπουμε οτι έχουμε βελτίωση χρόνου.

Για το τρίτο ερώτημα iii) έχουμε μα μικρή βελτίωση χρόνου από όταν χρησιμοποιούμε 16 parallel workers. Βέβαια δεν έχουμε την ίδια βελτίωση χρόνου σε σχέση με όταν ήμασταν μέσα στην ram και όταν έχουμε shared buffers 16GB.

Για το τέταρτο ερώτημα iv) έχουμε βελτίωση χρόνου από όλα τα υπόλοιπα ερωτήματα ακόμα και από όταν βρισκόμαστε μέσα στην ram.Έχουμε χρόνο 312msec 578 μέσα στην ram,έχουμε ρίξει τον χρόνο στο ήμισυ.

Για το πέμπτο ερώτημα v) έχουμε βελτίωση χρόνου από όλα τα υπόλοιπα ερωτήματα ακόμα και από όταν βρισκόμαστε μέσα στην ram. Έχουμε χρόνο 1.2sec ένω 1.5msec μέσα στην ram,έχουμε ρίξει το χρόνο 0.3sec.

Ερώτημα 2 (40%). Υλοποίηση Ερωτημάτων σε Spark (RDD)

Αφού φορτώσετε το σύνολο δεδομένων που σας δίνεται στο Spark:

Για να φωρτόσουμε τα δεδομένα αρχικά κάνουμε εγκατάσταση το eclipse.Το eclipse θα το κάνουμε εγκατάσταση σε kali linux.Έχω κάνει εγκατάσταση το μηχάνημα σε virtual box.



Κατεβάζω από την ιστοσελίδα του μαθήματος το αντίστοιχο virtual box και το hadoop για τις βιβλιοθήκες του spark.

The screenshot shows a Firefox browser window with the following details:

- Address Bar:** https://gUNET2.cs.unipi.gr/modules/announcements/announcements.php?course=TMCL110&an_id=14102
- Page Title:** GUNet2 eClass - Τμήμα Πληροφορικής | ΣΥΣΤΗΜΑΤΑ ΔΙΑΧΕΙΡΙΣΗΣ ΒΑΣΕΩΝ ΔΕΔΟΜΕΝΩΝ (5ο εξ.) | Ανακοινώσεις | Σειρά Εργαστηριακή Διάλεξη - Mozilla Firefox
- Content:** The main content area shows course announcements for 'ΣΥΣΤΗΜΑΤΑ ΔΙΑΧΕΙΡΙΣΗΣ ΒΑΣΕΩΝ ΔΕΔΟΜΕΝΩΝ (5ο εξ.)'. It includes links to download Java 8 and Eclipse 2018-09 versions.
- Bottom Right:** Powered by OPENeCLASS

The screenshot shows a Firefox browser window with the following details:

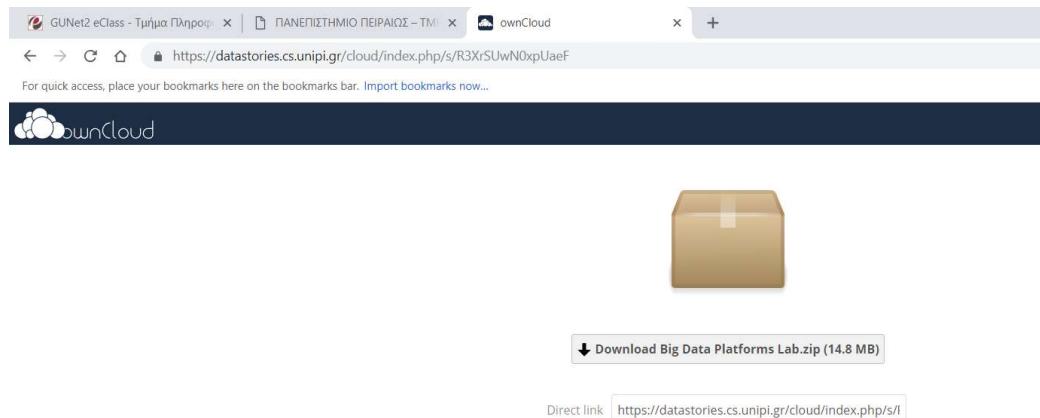
- Address Bar:** https://www.eclipse.org/downloads/download.php?file=/technology/epp/downloads/release/2018-09/R/eclipse-java-2018-09-linux-gtk-x86_64.tar.gz
- Page Title:** Eclipse downloads - Select a mirror | The Eclipse Foundation - Mozilla Firefox
- Content:** The page shows the download link for 'eclipse-java-2018-09-linux-gtk-x86_64.tar.gz' from the University of Erlangen-Nuremberg.
- Bottom Right:** Powered by OPENeCLASS

Κατεβάζουμε το αρχείο και κάνουμε εγκατάσταση το eclipse.

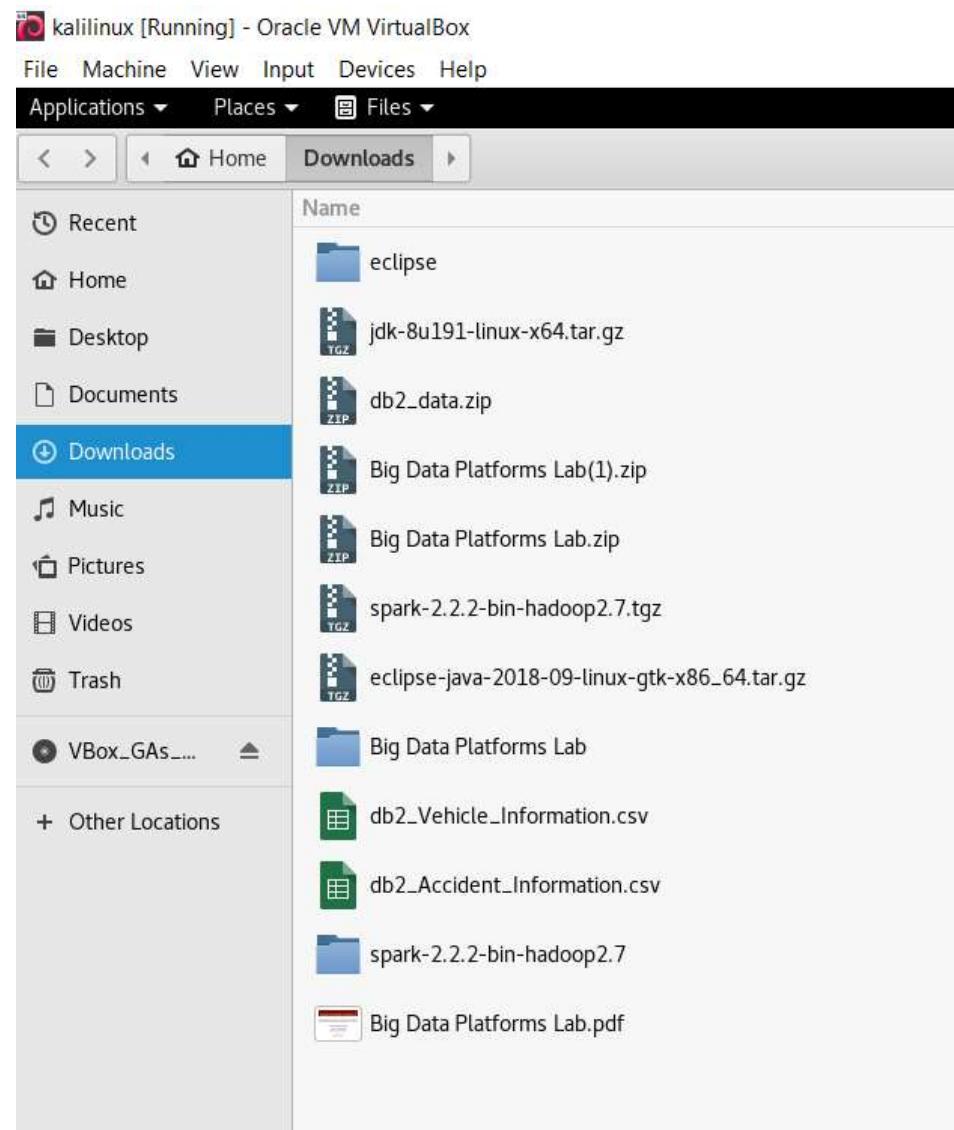
Στην συνέχεια κατεβάζουμε τις βιβλιοθήκες μας για το spark.

Επίσης έχουμε κάνει εγκατάσταση και την java έκδοση 1.8

Τέλος κατεβάζουμε και τον κώδικα που μας έχει δοθεί από το εργαστήριο.

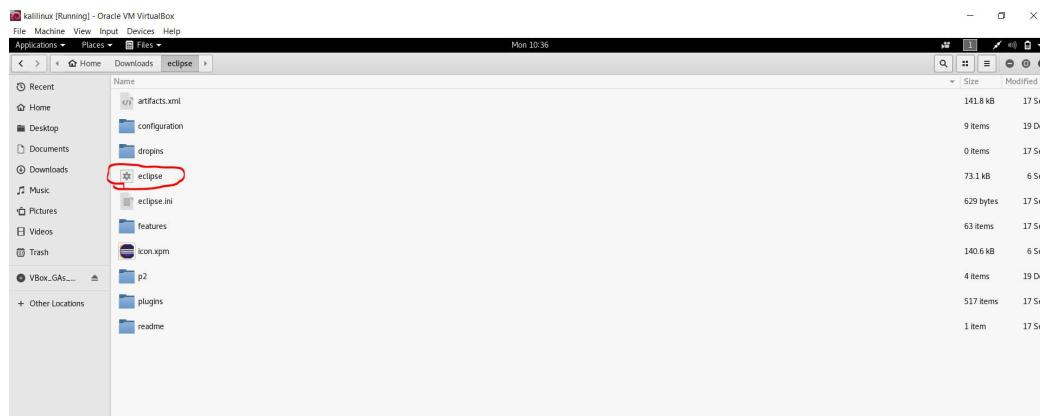


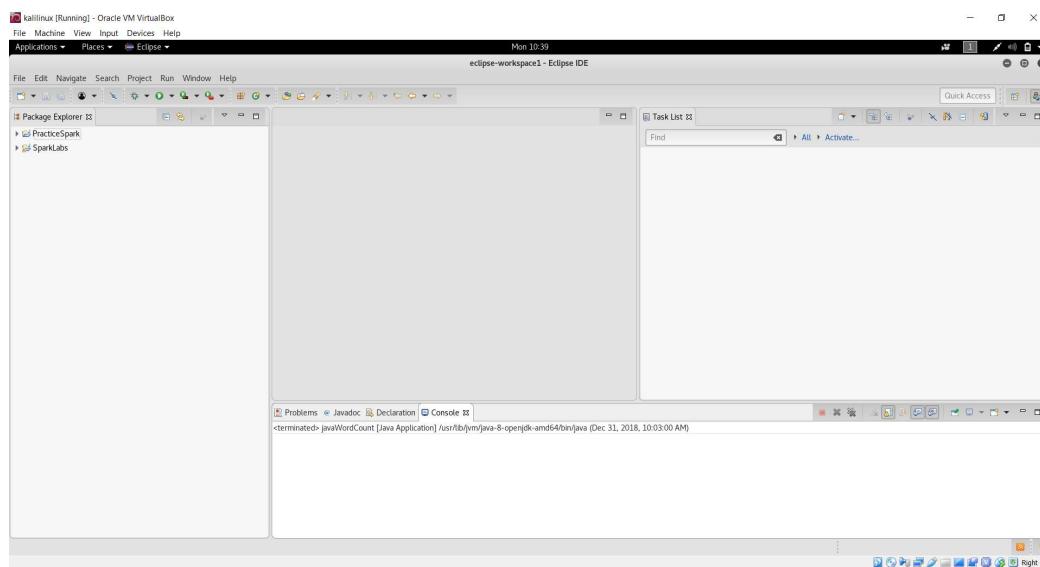
The screenshot shows a web browser window with three tabs open: GUNet2 eClass - Τμήμα Πληροφορικής, ΠΑΝΕΠΙΣΤΗΜΙΟ ΠΕΙΡΑΙΩΣ – ΤΜΠ, and ownCloud. The URL in the address bar is <https://datastories.cs.unipi.gr/cloud/index.php/s/R3XrSUwN0xpUaeF>. Below the address bar, there is a message: "For quick access, place your bookmarks here on the bookmarks bar. Import bookmarks now...". The main content area shows a brown cardboard box icon representing a download. A button below it says "Download Big Data Platforms Lab.zip (14.8 MB)". At the bottom, there is a link labeled "Direct link" followed by the same URL: <https://datastories.cs.unipi.gr/cloud/index.php/s/l>.



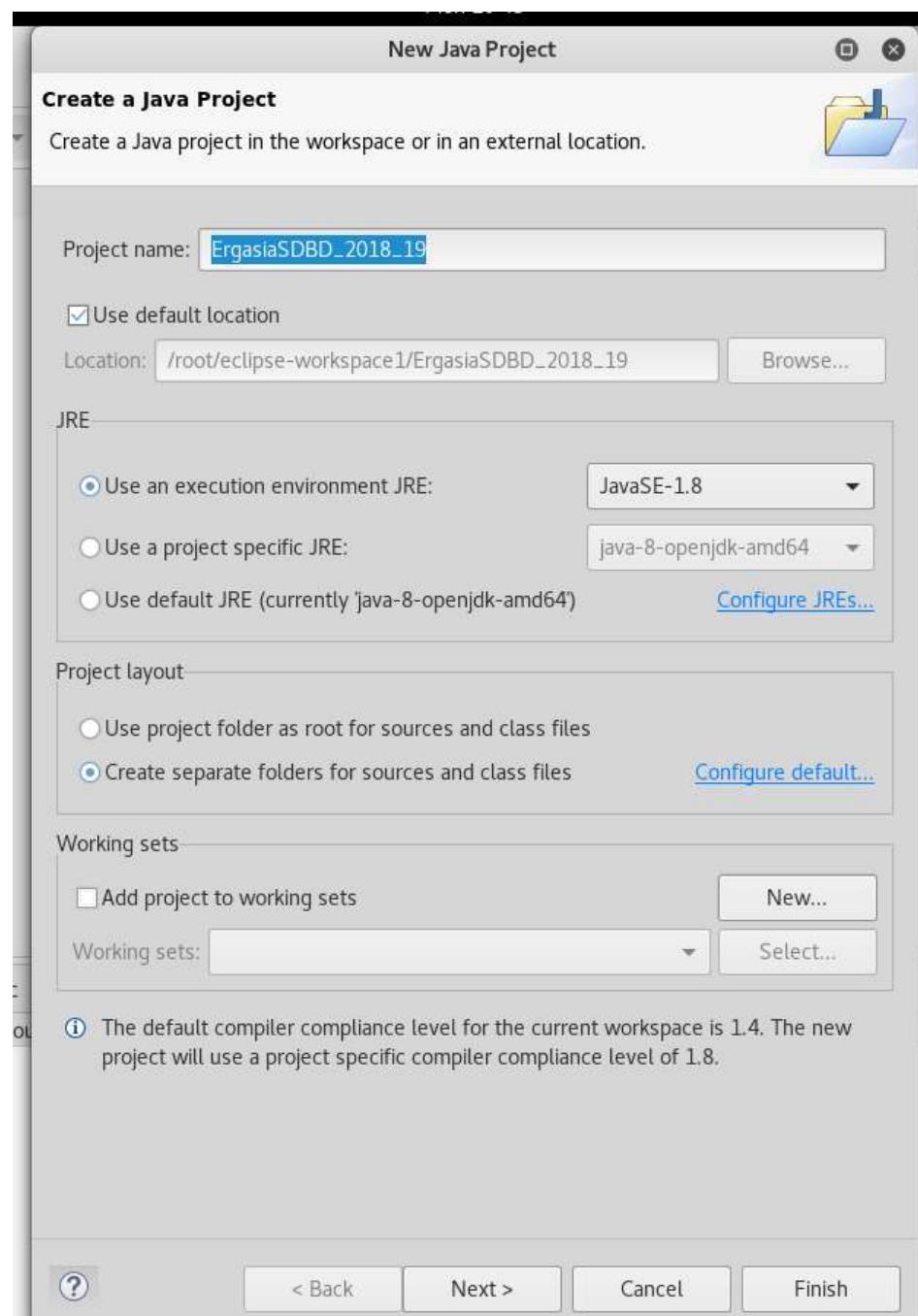
Και τέλος κατεβάζουμε και τα δυο αρχεία με τα οποία δουλεύουμε.

Για να ανοίξουμε το eclipse ανοίγουμε στον αντίστοιχο φάκελο και πατάμε αριστερό κλικ στο eclipse.exe

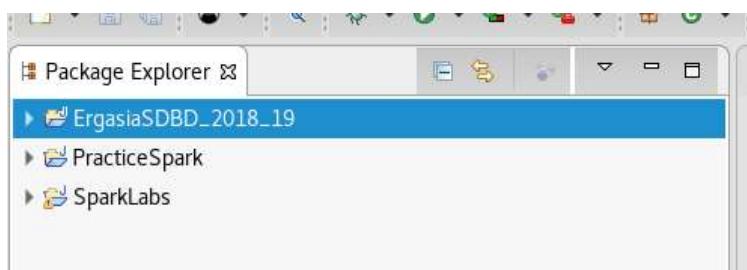


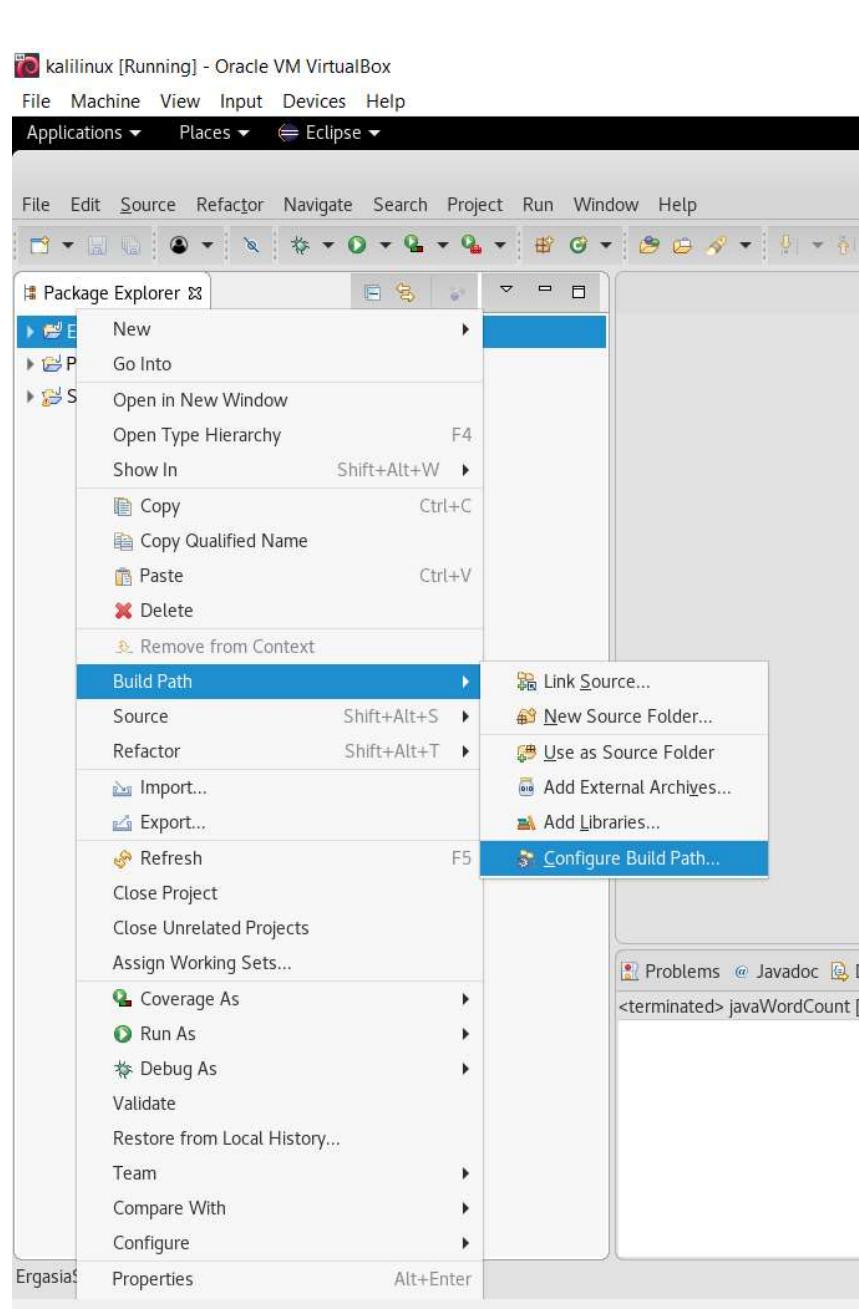


Δημιουργώ ένα νέο Project με όνομα **ErgasiaSDB_2018_19**



To project μας έχει δημιουργηθεί



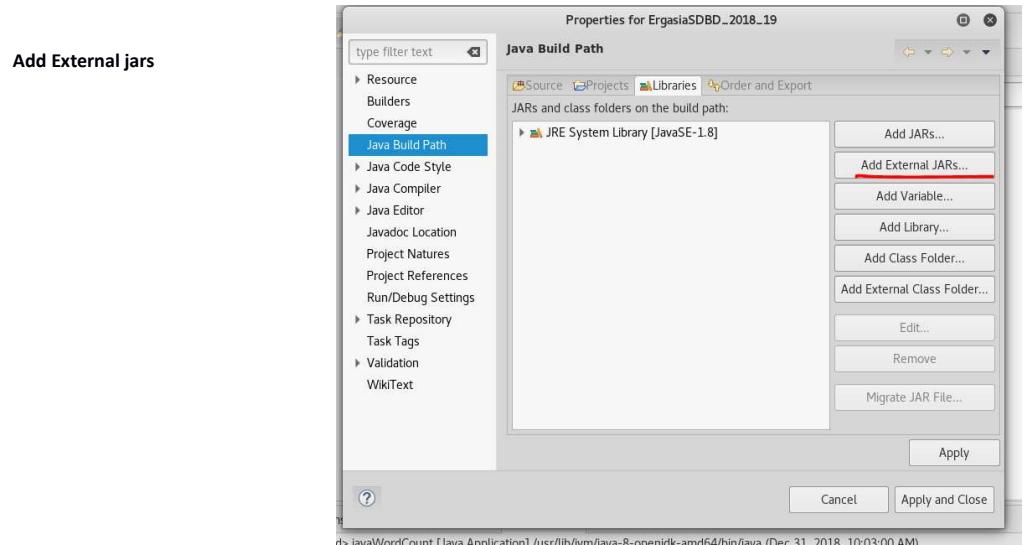


Κάνουμε δεξιά κλικ

Build Path

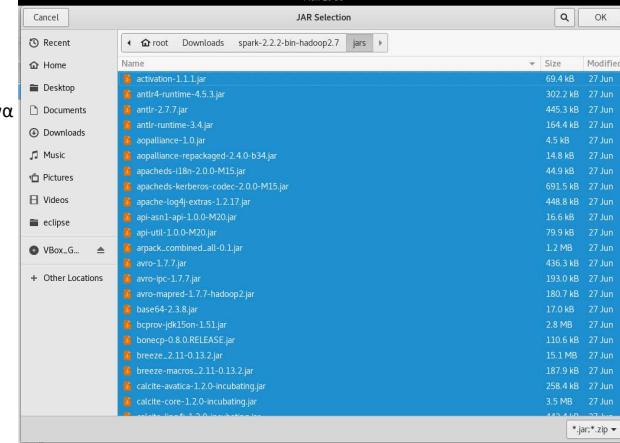
→Configure Build

Path

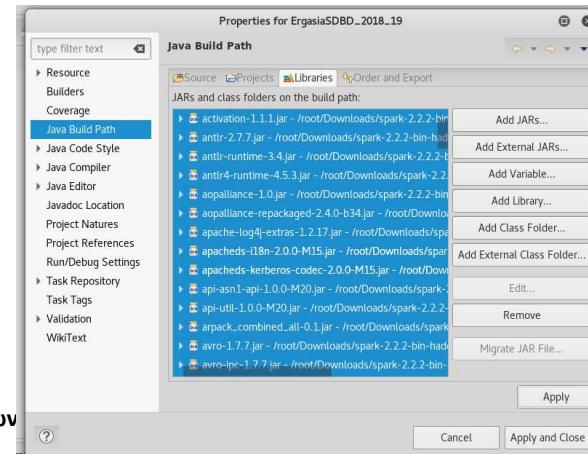


```
d$ javaWordCount Java Application1 /usr/lib/jvm/java-8-openjdk-amd64/bin/java (Dec 31, 2018, 10:03:00 AM)
```

Επιλέγουμε όλες τις βιβλιοθήκες μας και πατάμε **OK** από το αρχείο που κατεβάσαμε προηγούμενος για να κατεβάσουμε τις βιβλιοθήκες από το spark.



Τέλος πατάμε **apply** για να κάνουμε save στις αλλαγές μας.



Συστήματα Διαχείρισης Βάσεων Δεδομένων

(a) Υπολογίστε κάποια στατιστικά για τα δεδομένα (ενδεικτικά, μπορείτε να υπολογίσετε πόσοι είναι οι κατασκευαστές των αυτοκινήτων, πόσα ατυχήματα έγιναν ανά έτος, κλπ.).

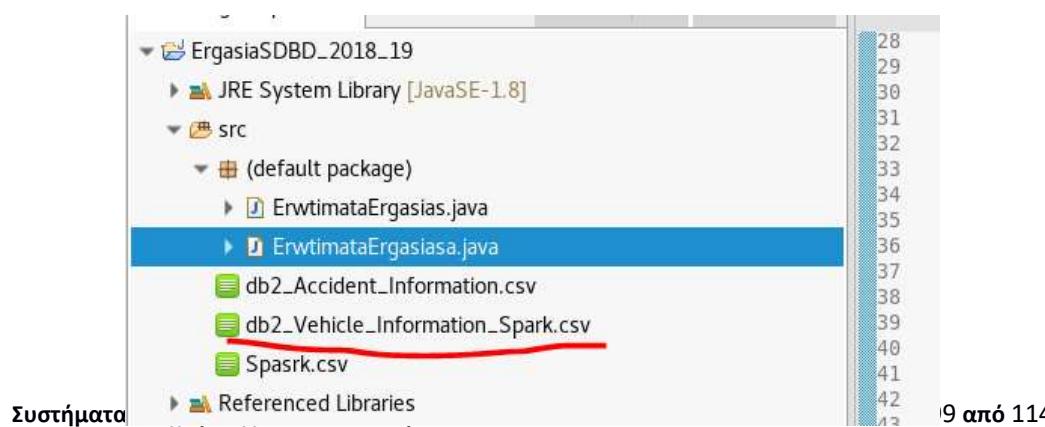
Θα υπολογίσω πόσοι είναι οι κατασκευαστές των αυτοκινήτων.

Θα χρειαστώ το δεύτερο αρχείο db2_Vehicle_Information.csv. Και για να βρω πόσοι είναι οι κατασκευαστές των αυτοκινήτων και θα πάρω την στήλη **make**.

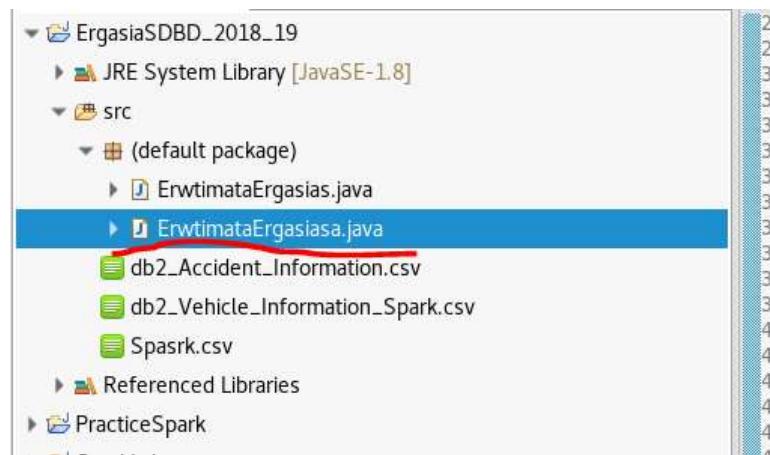
E	F	G	H	I	J	K
Vehicle make	model	Sex_of_Driver	Vehicle_Type			
3 ROVER	45 CLASSIC	Male	109			
BMW	C1	Male	109			
4 NISSAN	MICRA CEI	Male	109			
LONDON TAXIS INT	TXII GOLD	Male	109			
1 PIAGGIO	VESPA ET4	Male	Motorcycle 125cc and under			
10 VOLKSWAGEN		Male	109			
PIAGGIO	VESPA GT	Male	Motorcycle 125cc and under			
BMW	R1100 RT	Male	109			
3 MERCEDES		Male	109			
4 VOLKSWAGEN	GOLF V5	Female	109			
3 FORD	TRANSIT 3	Male	Van / Goods 3.5 tonnes mgw or under			
2 DENNIS		Male	Bus or coach (17 or more pass seats)			
11 VAUXHALL	CAVALIER	Male	109			
6 DENNIS		Not known	Bus or coach (17 or more pass seats)			
1 MERCEDES		Male	Goods 7.5 tonnes mgw and over			
it o MAN		Not known	Goods 7.5 tonnes mgw and over			
6 VOLKSWAGEN	GOLF GL A	Male	109			
PIAGGIO	VESPA ET4	Male	Motorcycle 125cc and under			
6 LONDON TAXIS INT	TX1 BRON	Female	108			
2 BMW	330 CI SPC	Male	109			
9 VOLVO		Male	Van / Goods 3.5 tonnes mgw or under			
5 FIAT	PUNTO 60	Male	109			
3 RENAULT	MEGANE S	Male	109			
13 AUDI	100E AUTO	Male	109			
11 MAZDA	XEDOS 6 S	Not known	109			
15 MERCEDES	500 SEL AL	Male	109			
11 ROVER	620 SI AUT	Male	109			
MAN		Male	Goods 7.5 tonnes mgw and over			

(+)

Παίρνω αυτό το αρχείο και το αποθηκεύω στο eclipse μου.



Στην συνέχεια δημιουργώ ένα .java αρχείο με όνομα **ErwtimataErgasiasa.java**



Στην συνέχεια θα κάνω ανάλυση του κώδικα μου.

```

File Edit Source Refactor Navigate Search Project Run Window Help
eclipse - Automatic suspend 2018_19/src/ErwtimataErgasiasa.java - Eclipse IDE
Computer will suspend very soon because of inactivity.

ErwtimataErgasiasa.java
11 public class ErwtimataErgasiasa {
12
13     private static final Pattern SPACE = Pattern.compile(" ");
14
15     public static void main(String[] args) {
16         // TODO Auto-generated method stub
17         SparkConf conf=new SparkConf();
18         conf.setAppName("javaWordCount").setMaster("local[*]");
19         JavaSparkContext jsc =new JavaSparkContext(conf);
20         JavaRDD<String>lines=jsc.textFile(System.getProperty("user.dir")+"/src/db2_Vehicle_Information_Spark.csv",4);
21
22         JavaRDD<String>words=lines.flatMap(s -> Arrays.asList(SPACE.split(s)).iterator());
23
24
25         //JavaPairRDD<String, Integer> ones = words.mapToPair(s -> new Tuple2<(s, 1));
26
27         JavaPairRDD<String, Integer> ones=words.mapToPair(s->{
28             String[] foo=s.split(",");
29
30             return new Tuple2<(foo[0],1);
31         });
32
33
34
35         JavaPairRDD<String, Integer> counts=ones.reduceByKey((i1,i2)-> i1 + i2 );
36
37         //System.out.println(words.first()+"hello");
38         //System.out.println(words.take(22));
39         int x=0;
40
41         List<Tuple2<String, Integer>> output = counts.collect();
42         for(Tuple2<?,?>tuple : output) {
43             System.out.println(tuple._1)+" "+ tuple._2());
44             x=x+1;
45         }
46         System.out.println("How many are the car manufacturers?");
47         System.out.println("The car manufacturers are:"+x);
48         jsc.close();
49     }
50
51 }
52
53 }
54

```

Αυτό που θέλω να πραγματοποιήσω σε αυτό το ερώτημα είναι να μετρήσω πόσοι είναι οι κατασκευαστές των αυτοκινήτων.

Οπότε αρχικά θα ανοίξω και θα διαβάζω γραμμή γραμμή το αρχείο που έχω **db2_Vehicle_Information_Spark.csv**

Το ανοίγω με την εντολή και διαβάζω το αρχείο μου γραμμή γραμμή:

```

JavaRDD<String>lines=jsc.textFile(System.getProperty("user.dir")+"/src/db2_Vehicle_Information_Spark.csv",4);

```

JavaRDD<String>lines=jsc.textFile(System.getProperty("user.dir")+"/src/db2_Vehicle_Information_Spark.csv",4);

Στην συνέχεια παίρνω την κάθε γραμμή και την βάζω μεσα σε μια λίστα σαν λέξεις.

```

JavaRDD<String>words=lines.flatMap(s -> Arrays.asList(SPACE.split(s)).iterator());

```

JavaRDD<String>words=lines.flatMap(s->Arrays.asList(SPACE.split(s)).iterator());

Στην συνέχεια παίρνω αυτές τις λέξεις και τις κάνω split με βάση το κόμμα

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

```
JavaPairRDD<String, Integer> ones =words.mapToPair(s->{
    String[] foo=s.split(",")
    και επειδή θέλω την πρώτη στήλη παίρνω το foo[0] και το κάνω count
    return new Tuple2<>(foo[0],1);
}
```

Στην συνέχεια κάνω count όλους του κατασκευαστές και τυπώνω πόσοι είναι.

```
JavaPairRDD<String, Integer> counts=ones.reduceByKey((i1,i2)-> i1 + i2 );
//System.out.println(words.first()+"hello");
//System.out.println(words.take(22));
int x=0;
List<Tuple2<String, Integer>> output = counts.collect();
for(Tuple2<?,?>tuple : output) {
    System.out.println(tuple._1()+": "+ tuple._2());
    x=x+1;
}
System.out.println("How many are the car manufacturers?");
System.out.println("The car manufacturers are:"+x);
jsc.close();
```

Εδώ τυπώνω αν κατασκευαστή πόσες φορές εμφανίστηκε.

System.out.println("tuple._1()"+":"+tuple._2());

Στο tuple._1() έχω τον κατασκευαστή και στο tuple._2() έχω πόσες φορές εμφανίζεται.

x=x+1;

System.out.println("How many are the car manufacturers?");

Εδώ τυπώνω πόσοι είναι οι κατασκευαστές.

System.out.println("The car manufacturers are:"+x);

```

1 // ErwimataErgasiasa.java
2
3 public static void main(String[] args) {
4     // TODO Auto-generated method stub
5     SparkConf conf=new SparkConf();
6     conf.setAppName("javawordCount").setMaster("local[*]");
7     JavaSparkContext jsc =new JavaSparkContext(conf);
8     JavaRDD<String> lines=jsc.textFile("user.dir"+"/src/db2_Vehicle_Information_Spark.csv",4);
9
10    JavaRDD<String> words=lines.flatMap(s -> Arrays.asList(SPACEx.split(s)).iterator());
11
12    JavaPairRDD<String, Integer> ones = words.mapToPair(s -> new Tuple2<(s, 1));
13
14    JavaPairRDD<String, Integer> ones=words.mapToPair(s->{
15        String[] foo=s.split(",");
16        return new Tuple2<(foo[0],1);
17    });
18
19    JavaPairRDD<String, Integer> counts=ones.reduceByKey((i1,i2)-> i1 + i2 );
20
21    System.out.println(words.first()+"hello");
22    System.out.println(words.take(22));
23    int x=0;
24
25    List<Tuple2<String, Integer>> output = counts.collect();
26    for(Tuple2<,?>tuple : output) {
27        System.out.println(tuple._1+": "+ tuple._2());
28        x=x+1;
29    }
30    System.out.println("How many are the car manufacturers?");
31    System.out.println("The car manufacturers are:"+x);
32    jsc.close();
33
34 }
35
36 }
37
38 }
39
40 }
41
42 }
43
44 }
45
46 }
47
48 }
49
50 }
51
52 }
53 }
54

```

Output window:

```

<terminated> ErwimataErgasiasa [Java Application] /usr/lib/jvm/jre
RANSOME: 36
LAROUS: 327
DARWIN: 4
WUYANG: 46
MOBYLETTE: 2
SWAP: 2
LML: 180
SCUTUM: 1
BENTLEY: 369
MURDO: 22
CF: 5
DS: 93
BENELLI: 91
INT.: 706
MIG: 2
SANBEN: 46
LUDVICO: 2657
TAYLOR: 2
XINTIAN: 3
BOOM: 8
ZENNCO: 10
SFWM: 1
JEFFREY: 2
HUMBER: 1
How many are the car manufacturers?
The car manufacturers are 501
19/01/02 18:15:00 INFO SparkUI: Stopped Spark web UI
19/01/02 18:15:00 INFO MapOutputTrackerMasterEndpoint
19/01/02 18:15:00 INFO MemoryStore: MemoryStore cleared
19/01/02 18:15:00 INFO BlockManager: BlockManager started
19/01/02 18:15:00 INFO BlockManagerMaster: BlockManagerMaster started
19/01/02 18:15:00 INFO OutputCommitCoordinator$Output
19/01/02 18:15:00 INFO SparkContext: Successfully started
19/01/02 18:15:00 INFO ShutdownHookManager: Shutdown hook registered
19/01/02 18:15:00 INFO ShutdownHookManager: Deleting

```

Όπως βλέπουμε τυπώνονται στην κονσόλα δεξιά πόσες φορές εμφανίζονται οι κατασκευαστές και στο τέλος αφού τους έχουμε μετρήσει με έναν counter **x=x+1;** εμφανίζω πόσοι είναι αυτοί οι κατασκευαστές εδώ είναι.

Οι κατασκευαστές των αυτοκινήτων είναι 581.

(b) Επιλέξτε ένα υποερώτημα από 1.a και προσαρμόστε το σε Spark με τη χρήση των κατάλληλων εντολών που παρέχονται από την πλατφόρμα (π.χ. map, reduce, reduceByKey, min, max, κλπ.).

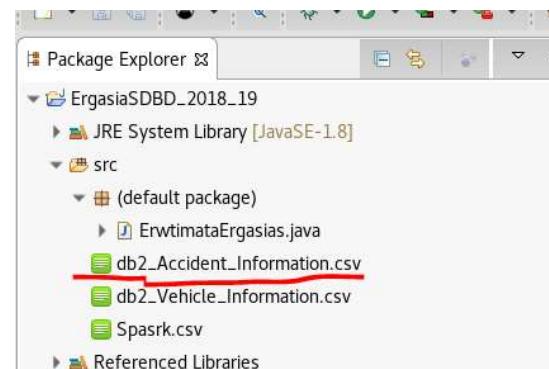
Θα πάρω το υποερώτημα 1.a

i. Βρείτε πόσα ατυχήματα έγιναν ανά κατηγορία δρόμου (road class).

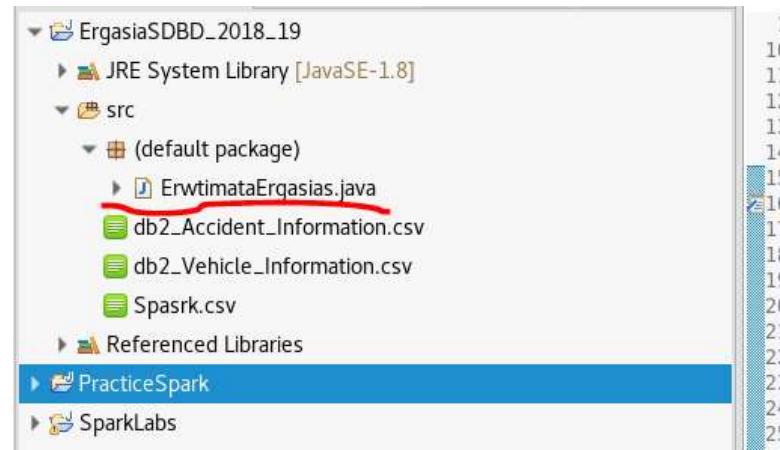
Για αυτό το ερώτημα θα χρειαστώ από τον πίνακα **db2_Accident_Information.csv** δύο στήλες την στήλη **accident_index** και την στήλη **first_road_class**.

A	B	C	D
1	Accident_Index	1st_Road_Class	Accident_Severity
2	0 200501BS00001	A	Serious
3	1 200501BS00002	B	Slight
4	2 200501BS00003	C	Slight
5	3 200501BS00004	A	Slight
6	4 200501BS00005	Unclassified	Slight
7	5 200501BS00006	Unclassified	Slight
8	6 200501BS00007	C	Slight
9	7 200501BS00009	A	Slight
10	8 200501BS00010	A	Slight
11	9 200501BS00011	B	Slight
12	10 200501BS00012	A	Slight
13	11 200501BS00014	A	Slight
14	12 200501BS00015	Unclassified	Slight
15	13 200501BS00016	A	Slight
16	14 200501BS00017	A	Slight
17	15 200501BS00018	A	Slight

Παίρνω αυτό το αρχείο και το αποθηκεύω στο eclipse μου.



Στην συνέχεια δημιουργώ ενα .java αρχείο με όνομα **ErwtimataErgasias.java**



Στην συνέχεια θα κάνω ανάλυση του κώδικα μου.

```

1  import java.util.Arrays;
2
3  public class ErwtimataErgasias {
4
5      private static final Pattern SPACE = Pattern.compile(" ");
6
7      public static void main(String[] args) {
8          // TODO Auto-generated method stub
9          SparkConf conf=new SparkConf();
10         conf.setAppName("avawordCount").setMaster("local[*]");
11         JavaSparkContext jsc =new JavaSparkContext(conf);
12         JavaRDD<String> lines=jsc.textFile(System.getProperty("user.dir")+"/src/db2_Accident_Information.csv",4);
13
14         JavaRDD<String> words=lines.flatMap(s -> Arrays.asList(SPACE.split(s)).iterator());
15
16         JavaPairRDD<String, Integer> ones = words.mapToPair(s -> new Tuple2<>(s, 1));
17
18         JavaPairRDD<String, Integer> ones=words.mapToPair(s->{
19             String[] foos=s.split(",");
20
21             return new Tuple2<>(foos[1],1);
22         });
23
24
25         JavaPairRDD<String, Integer> counts=ones.reduceByKey((i1,i2)-> i1 + i2 );
26
27         //System.out.println(words.first()+"hello");
28         //System.out.println(words.take(22));
29
30         List<Tuple2<String, Integer>> output = counts.collect();
31         for(Tuple2<?,?>tuple : output) {
32             System.out.println(tuple._1()+" "+ tuple._2());
33         }
34         jsc.close();
35     }
36
37 }
38
39
40
41
42
43
44
45
46
47
48
49 }
50

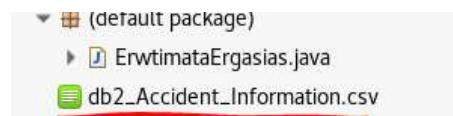
```

Αυτό που θέλω να πραγματοποιήσω σε αυτό το ερώτημα είναι να μετρήσω την στήλη accident index με βάση την στήλη first_road_class ομαδοποιημένη.

Οπότε αρχικά θα ανοίξω και θα διαβάσω γραμμή γραμμή το αρχείο που έχω db2_Accident_Information.csv

Το ανοίγω με την εντολή και διαβάζω το αρχείο μου γραμμή γραμμή:

```
JavaRDD<String>lines=jsc.textFile(System.getProperty("user.dir")+"/src/db2_Accident_Information.csv",4);
JavaRDD<String>lines=jsc.textFile(System.getProperty("user.dir")+"src/db2_Accident_Information.csv",4);
```



Στην συνέχεια παίρνω την κάθε γραμμή και την βάζω μέσα σε μια λίστα σαν λέξεις.

```
JavaRDD<String>words=lines.flatMap(s -> Arrays.asList(SPACEx.split(s)).iterator());
```

JavaRDD<String>words=lines.flatMap(s->Arrays.asList(SPACEx.split(s)).iterator());

Στην συνέχεια παίρνω αυτές τις λέξεις και τις κάνω split με βάση το κομμα

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

JavaPairRDD<String, Integer> ones =words.mapToPair(s->{

String[] foo=s.split(",")

και επειδή θέλω την δεύτερη στήλη παίρνω το foo[1] δηλαδή την στήλη first_road_class και την κάνω count

return new Tuple2<>(foo[1],1);

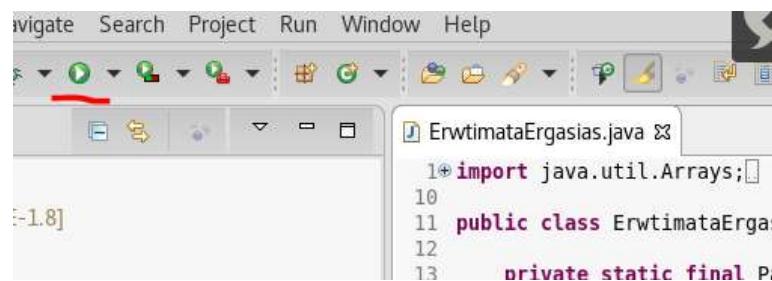
}

Στην συνέχεια κάνω count τις στήλες με βάση την στήλη first_road_class και τυπώνω τα αποτελέσματα.

```
JavaPairRDD<String, Integer> counts=ones.reduceByKey((i1,i2)-> i1 + i2 );
//System.out.println(words.first()+"hello");
//System.out.println(words.take(22));

List<Tuple2<String, Integer>> output = counts.collect();
for(Tuple2<?,?>tuple : output) {
    System.out.println(tuple._1()+": "+ tuple._2());
}
jsc.close();
```

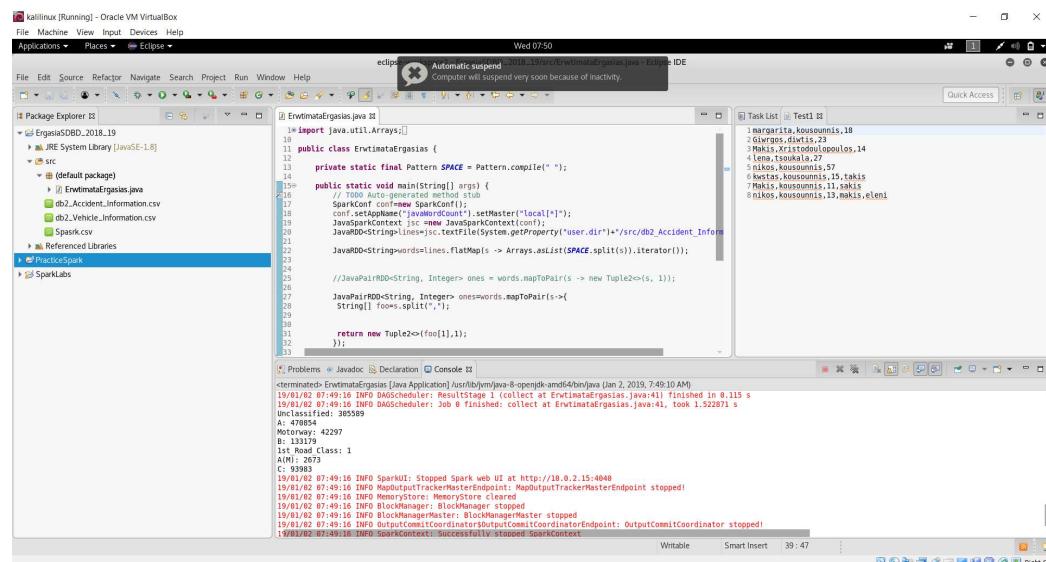
Πατάω play για να τρέξει το πρόγραμμα και βλέπω τα αποτελέσματα.



```

Navigate Search Project Run Window Help
File Machine View Input Devices Help
Applications Places Eclipse
File Edit Source Refactor Navigate Search Project Run Window Help
[1.8]
ErwtimataErgasias.java
import java.util.Arrays;
public class ErwtimataErgas
private static final Pa

```



```

kallinux [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Applications Places Eclipse
File Edit Source Refactor Navigate Search Project Run Window Help
[1.8]
ErwtimataErgasias.java
Wed 07:50
eclipse Automatic suspend 2018-11-07/ErwtimataErgasias.java-Eclipse IDE
Computer will suspend very soon because of inactivity.
File Edit Source Refactor Navigate Search Project Run Window Help
Package Explorer Task List Test1
src
  ▾ (default package)
    ▾ ErwtimataErgasias.java
      db2_Accident_Information.csv
      db2_Vehicle_Information.csv
      SparkCSV
    Referenced Libraries
  PracticeSpark
  SparkLabs
  Problems Declaration Console
<com.mkyong-ErwtimataErgasias [Java Application]>/lib/binary/jar-8-sparklib-ml-0.4.0.jar [jar 2, 2019-7-4 10:10 AM]
19/8/2019 07:49:16 INFO DAGScheduler: ResultStage 1 (collect at ErwtimataErgasias.java:41) finished in 0.119 s
19/8/2019 07:49:16 INFO DAGScheduler: Job 8 finished: collect at ErwtimataErgasias.java:41, took 1.522871 s
Unclassified: 385589
Motorway: 42297
B: 133179
M: 122222
A(M): 2673
C: 93983
19/8/2019 07:49:16 INFO SparkUI: Stopped Spark web UI at http://10.0.2.15:4040
19/8/2019 07:49:16 INFO MapOutputTrackerMasterEndpoint: MapOutputTrackerMasterEndpoint stopped!
19/8/2019 07:49:16 INFO MemoryStore: MemoryStore cleared
19/8/2019 07:49:16 INFO BlockManagerMaster: BlockManagerMaster stopped
19/8/2019 07:49:16 INFO BlockManager: BlockManager stopped
19/8/2019 07:49:16 INFO OutputCommitCoordinator$OutputCommitCoordinatorEndpoint: OutputCommitCoordinator stopped!
19/8/2019 07:49:16 INFO SparkContext: Successfully Stopped SparkContext

```

```
<terminated> ErwtimataErgasias [Java Application] /usr/lib/jvm/java-8-openjdk-amd64/bin/java (Jan 2, 2019, 7:49:10 AM)
19/01/02 07:49:16 INFO DAGScheduler: ResultStage 1 (collect at ErwtimataErgasias.java:41) finished in 0.115 s
19/01/02 07:49:16 INFO DAGScheduler: Job 0 finished: collect at ErwtimataErgasias.java:41, took 1.522871 s
Unclassified: 305589
A: 470854
Motorway: 42297
B: 133179
1st_Road_Class: 1
A(M): 2673
C: 93983
19/01/02 07:49:16 INFO SparkUI: Stopped Spark web UI at http://10.0.2.15:4040
19/01/02 07:49:16 INFO MapOutputTrackerMasterEndpoint: MapOutputTrackerMasterEndpoint stopped!
19/01/02 07:49:16 INFO MemoryStore: MemoryStore cleared
19/01/02 07:49:16 INFO BlockManager: BlockManager stopped
19/01/02 07:49:16 INFO BlockManagerMaster: BlockManagerMaster stopped
19/01/02 07:49:16 INFO OutputCommitCoordinator$OutputCommitCoordinatorEndpoint: OutputCommitCoordinator stopped!
19/01/02 07:49:16 INFO SparkContext: Successfully stopped SparkContext
```

Writable Smart Insert 39 : 47

Βλέπουμε ότι μας εμφανίζει δύο στήλες την στήλη **first_road_class** και την στήλη **count(accident_index)**.

Unclassified: 305589

A: 470854

Motorway: 42297

B:133179

1st_Road_class:1

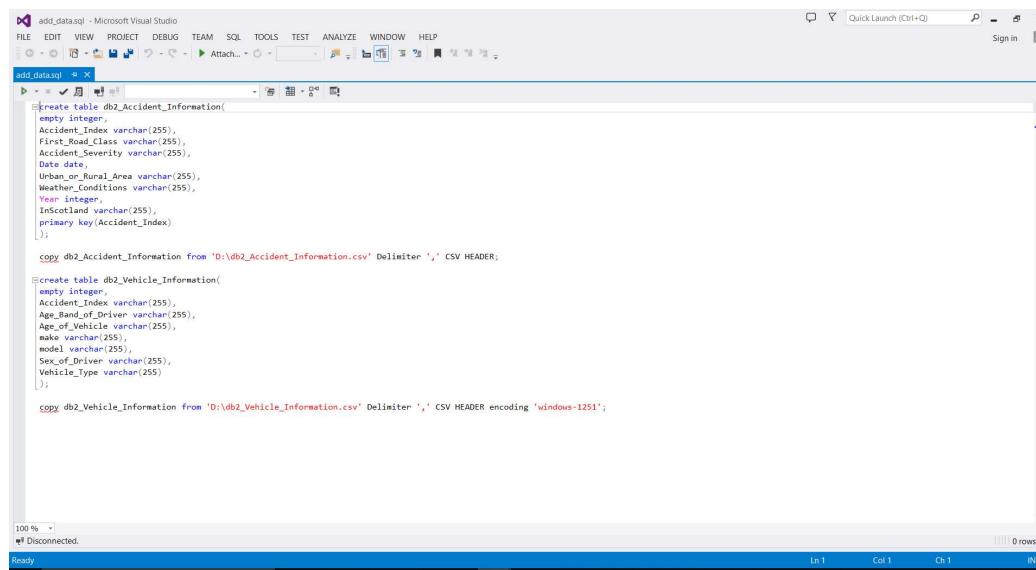
A(M): 2673

C: 93983

Μας εμφανίζει πόσα ατυχήματα έγιναν ανά κατηγορία δρόμου.

Κώδικας Όλης της Εργασίας

Συστήματα Διαχείρισης Βάσεων Δεδομένων 2018-19 (5ο εξάμηνο)



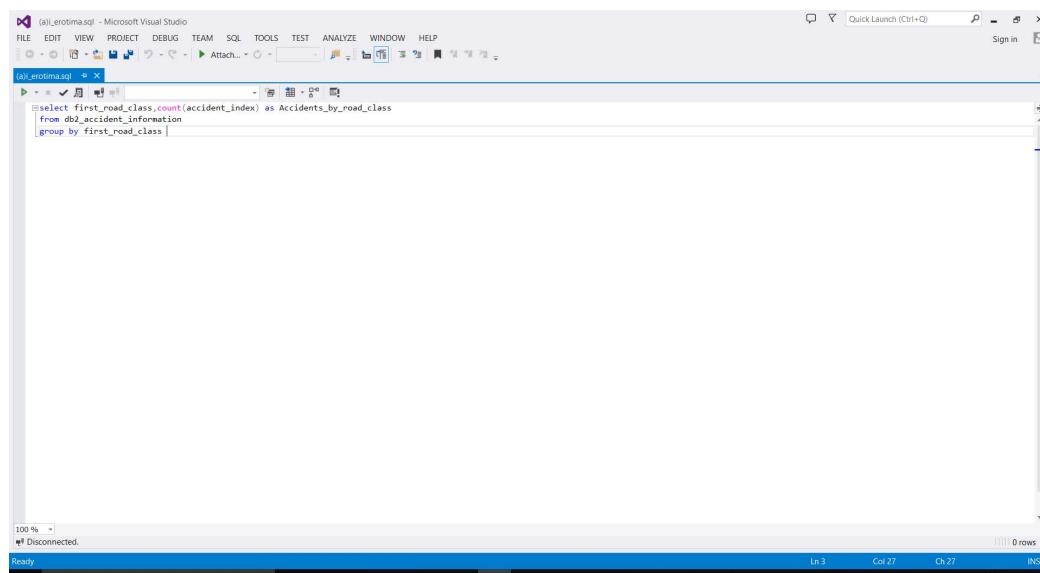
```
add_data.sql - Microsoft Visual Studio
FILE EDIT VIEW PROJECT DEBUG TEAM SQL TOOLS TEST ANALYZE WINDOW HELP
Quick Launch (Ctrl+Q) Sign in
add_data.sql
CREATE TABLE db2_Accident_Information(
    empty integer,
    Accident_Index varchar(255),
    First_Road_Class varchar(255),
    Accident_Severity varchar(255),
    Date date,
    Urban_or_Rural_Area varchar(255),
    Weather_Conditions varchar(255),
    Year integer,
    InScotland varchar(255),
    primary key(Accident_Index)
);

COPY db2_Accident_Information FROM 'D:\db2_Accident_Information.csv' Delimiter ',' CSV HEADER;

CREATE TABLE db2_Vehicle_Information(
    empty integer,
    Accident_Index varchar(255),
    Age_Band_of_Driver varchar(255),
    Age_of_Vehicle varchar(255),
    make varchar(255),
    model varchar(255),
    Sex_of_Driver varchar(255),
    Vehicle_Type varchar(255)
);

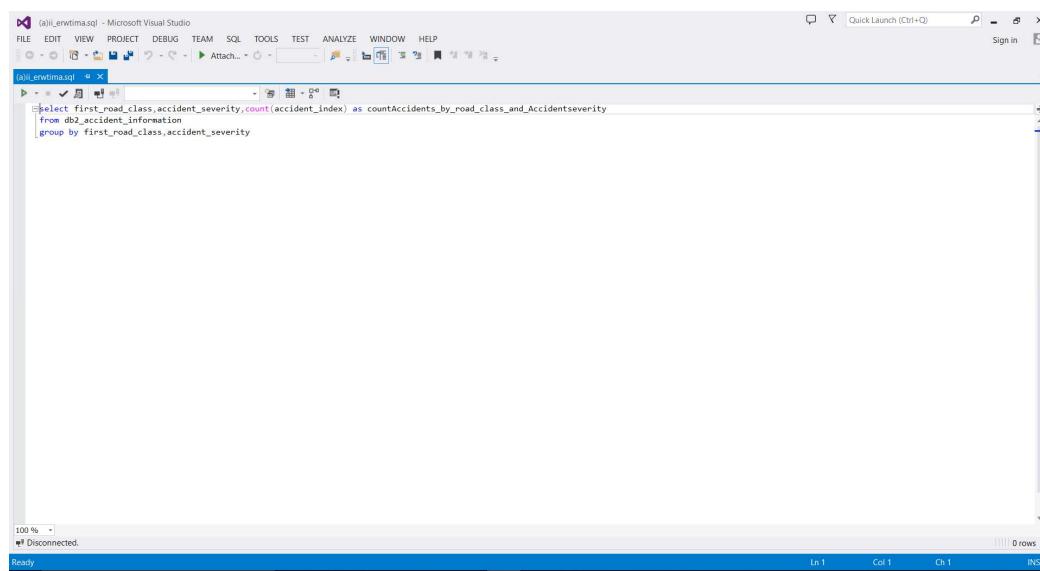
COPY db2_Vehicle_Information FROM 'D:\db2_Vehicle_Information.csv' Delimiter ',' CSV HEADER encoding 'windows-1251';

100 % | Disconnected.
0 rows
Ln 1 Col 1 Ch 1 INS
Ready
```



```
(a)_erottima.sql - Microsoft Visual Studio
FILE EDIT VIEW PROJECT DEBUG TEAM SQL TOOLS TEST ANALYZE WINDOW HELP
(a)_erottima.sql 0 rows
select first_road_class,count(accident_index) as Accidents_by_road_class
from db2_accident_information
group by first_road_class

100 % 0 rows
Disconnected.
Ready Col 1 Ch 1 INS
```



```
(a)_erwtima.sql - Microsoft Visual Studio
FILE EDIT VIEW PROJECT DEBUG TEAM SQL TOOLS TEST ANALYZE WINDOW HELP
(a)_erwtima.sql 0 rows
select first_road_class,accident_severity,count(accident_index) as countAccidents_by_road_class_and_Accidentseverity
from db2_accident_information
group by first_road_class,accident_severity

100 % 0 rows
Disconnected.
Ready Col 1 Ch 1 INS
```



The screenshot shows the Microsoft Visual Studio interface with the following details:

- Title Bar:** (aqli_eroitma.sql) - Microsoft Visual Studio
- Menu Bar:** FILE EDIT VIEW PROJECT DEBUG TEAM SQL TOOLS TEST ANALYZE WINDOW HELP
- Toolbar:** Includes icons for New, Open, Save, Print, Find, Copy, Paste, Cut, Undo, Redo, and others.
- Code Editor:** The file 'aqli_eroitma.sql' is open, displaying the following SQL query:

```
select count(db2_accident_information.accident_index) as UrbanAccidents_less2010_1_1_agebetween2635
from db2_accident_information
inner join db2_Vehicle_Information
on db2_accident_information.accident_index=db2_Vehicle_Information.accident_index
where db2_accident_information.urban_or_rural_area='Urban' and db2_accident_information.date<'2010/1/1' and db2_Vehicle_Information.age_band_of_driver='26 - 35'
```
- Status Bar:** Shows '100 %' and 'Disconnected.'
- Bottom Right Corner:** Shows '0 rows' and 'INS' (Insert mode).

The screenshot shows the Microsoft Visual Studio interface with the following details:

- Title Bar:** (alv_erottama.sql) * - Microsoft Visual Studio
- Menu Bar:** FILE EDIT VIEW PROJECT DEBUG TEAM SQL TOOLS TEST ANALYZE WINDOW HELP
- Toolbar:** Standard toolbar icons for opening, saving, and running queries.
- Quick Launch:** Quick Launch (Ctrl+Q) and Sign in button.
- Code Editor:** The file 'alv_erottama.sql' is open, containing the following SQL query:

```
select count(db2_accident_information.accident_index) as RuralAccidents_Year2012_agebetween3645
from db2_accident_information
inner join db2_Vehicle_Information
on db2_accident_information.accident_index=db2_Vehicle_Information.accident_index
where db2_accident_information.urban_or_rural_area='Rural' and db2_accident_information.date>'2012/1/1' and db2_accident_information.date<'2012/12/31'
and db2_Vehicle_Information.age_band_of_driver='36 - 45'
```
- Status Bar:** 100 %, Disconnected, Ready, Line 6, Column 1, Ch 1, INS.

(alv_eroitma.sql) - Microsoft Visual Studio

FILE EDIT VIEW PROJECT DEBUG TEAM SQL TOOLS TEST ANALYZE WINDOW HELP

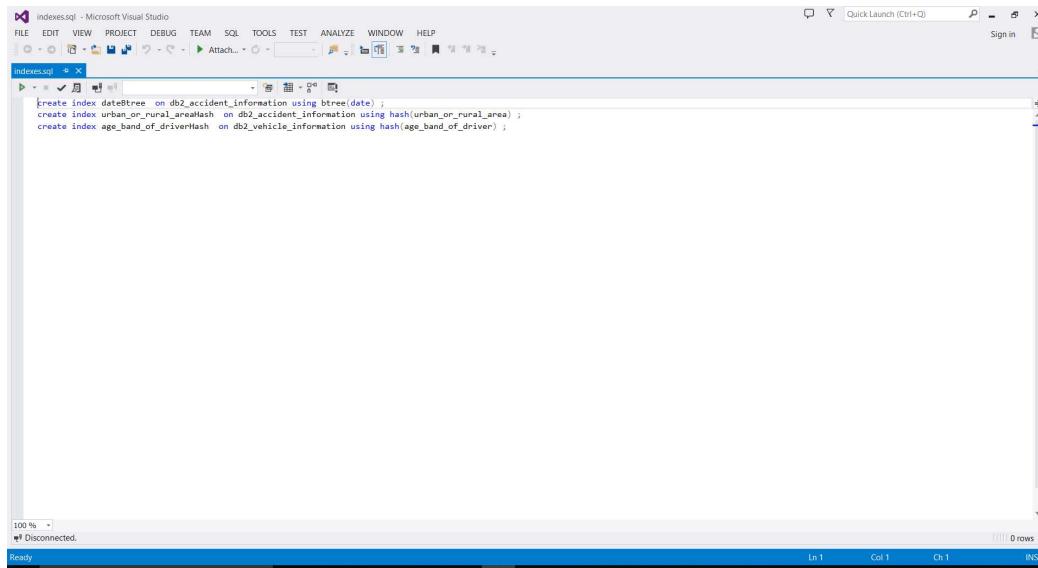
Quick Launch (Ctrl+Q)

Sign In

alv_eroitma.sql

```
select db2_vehicle_information.make as manufacturer, count(db2_vehicle_information.make) maxAccidents
from db2_vehicle_information
inner join db2_accident_information
on db2_vehicle_information.accident_index=db2_accident_information.accident_index
where db2_accident_information.date<='2010/1/1' and db2_accident_information.date<='2012/12/31' and db2_vehicle_information.Age_Band_of_Driver='26 - 35' and db2_accident_information.first_road_class='A'
group by db2_vehicle_information.make
having count(db2_vehicle_information.make)<(
select max(maxAccidents)
from select db2_vehicle_information.make as manufacturer, count(db2_vehicle_information.make)as count
from db2_vehicle_information
inner join db2_accident_information
on db2_vehicle_information.accident_index=db2_accident_information.accident_index
where db2_accident_information.date>='2010/1/1' and db2_accident_information.date<='2012/12/31' and db2_vehicle_information.Age_Band_of_Driver='26 - 35' and db2_accident_information.first_road_class='A'
group by manufacturer) as manufacturerType;
```

--brisku tous kataskeyastes kai posa atumxata exou kanei kai sthn synxeia basu to iso(having count(db2_vehicle_information.make)=)
--gia na brw to megisto apo aytous toys arithmos pou to brisku me to na ksesa diourghsou ton idio pinaka kataskeyastes kai posa atumxata gia na brw parw to ayton ton
--pinaka polo enihi to megisto wslas kai de afinei na emfanisels alles stiles den ton tis omopoihsis dhdhan erithika to megisto alla emfanize mono ton arithmos
--mono toy kai ton kataskeusti apote le basi ths sthn ths kataskeusti kai posa atumxata exou brisku mega apo ton sthn kai posa atumxata exou brisku arithmos

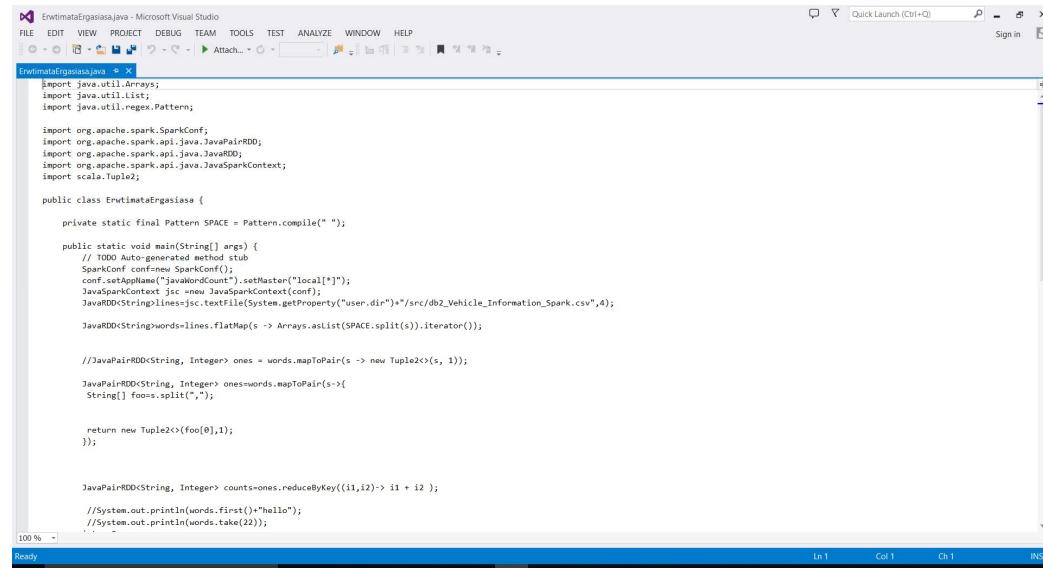


```
indexes.sql - Microsoft Visual Studio
FILE EDIT VIEW PROJECT DEBUG TEAM SQL TOOLS TEST ANALYZE WINDOW HELP
Sign in
indexes.sql > X
CREATE INDEX dateBtree ON db2_accident_information USING BTREE(date);
CREATE INDEX urban_or_rural_areaHash ON db2_accident_information USING HASH(urban_or_rural_area);
CREATE INDEX age_band_of_driverHash ON db2_vehicle_information USING HASH(age_band_of_driver);
```

100 % | 0 rows

Disconnected.

Ready Col 1 Col 2 Col 3 INS



```

ErwtimataErgasia.java - Microsoft Visual Studio
FILE EDIT VIEW PROJECT DEBUG TEAM TOOLS TEST ANALYZE WINDOW HELP
Quick Launch (Ctrl+Q) Sign in
ErwtimataErgasia.java
import java.util.Arrays;
import java.util.List;
import java.util.regex.Pattern;

import org.apache.spark.SparkConf;
import org.apache.spark.api.java.JavaPairRDD;
import org.apache.spark.api.java.JavaRDD;
import org.apache.spark.api.java.JavaSparkContext;
import scala.Tuple2;

public class ErwtimataErgasia {
    private static final Pattern SPACE = Pattern.compile(" ");

    public static void main(String[] args) {
        // TODO Auto-generated method stub
        SparkConf conf=new SparkConf();
        conf.setAppName("JavaWordCount");
        conf.setMaster("local[*]");
        JavaSparkContext jsc = new JavaSparkContext(conf);
        JavaRDD<String> lines=jsc.textFile(System.getProperty("user.dir")+"/src/db2_Vehicle_Information_Spark.csv",4);

        JavaRDD<String> words=lines.flatMap(s-> Arrays.asList(SPACE.split(s)).iterator());

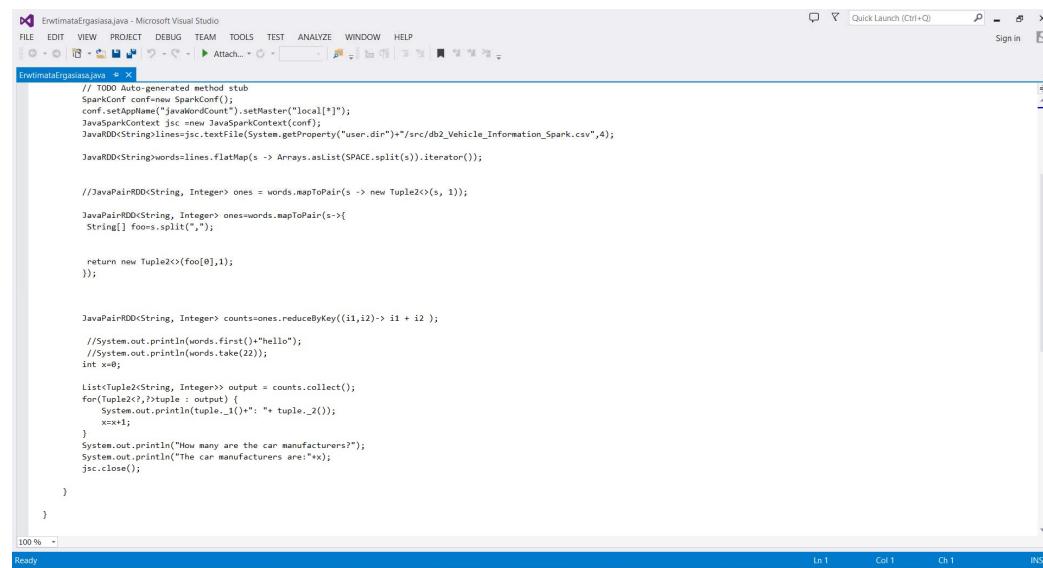
        //JavaPairRDD<String, Integer> ones = words.mapToPair(s-> new Tuple2<s, 1>);

        JavaPairRDD<String, Integer> ones=words.mapToPair(s->{
            String[] foons=split(",");
            return new Tuple2<>(foons[0],1);
        });

        JavaPairRDD<String, Integer> counts=ones.reduceByKey((i1,i2)-> i1 + i2 );

        //System.out.println(words.first()+"hello");
        //System.out.println(words.take(22));
    }
}

```



```

ErwtimataErgasia.java
FILE EDIT VIEW PROJECT DEBUG TEAM TOOLS TEST ANALYZE WINDOW HELP
Quick Launch (Ctrl+Q) Sign in
ErwtimataErgasia.java
import java.util.Arrays;
import java.util.List;
import java.util.regex.Pattern;

import org.apache.spark.SparkConf;
import org.apache.spark.api.java.JavaPairRDD;
import org.apache.spark.api.java.JavaRDD;
import org.apache.spark.api.java.JavaSparkContext;
import scala.Tuple2;

public class ErwtimataErgasia {
    private static final Pattern SPACE = Pattern.compile(" ");

    public static void main(String[] args) {
        // TODO Auto-generated method stub
        SparkConf conf=new SparkConf();
        conf.setAppName("JavaWordCount");
        conf.setMaster("local[*]");
        JavaSparkContext jsc = new JavaSparkContext(conf);
        JavaRDD<String> lines=jsc.textFile(System.getProperty("user.dir")+"/src/db2_Vehicle_Information_Spark.csv",4);

        JavaRDD<String> words=lines.flatMap(s-> Arrays.asList(SPACE.split(s)).iterator());

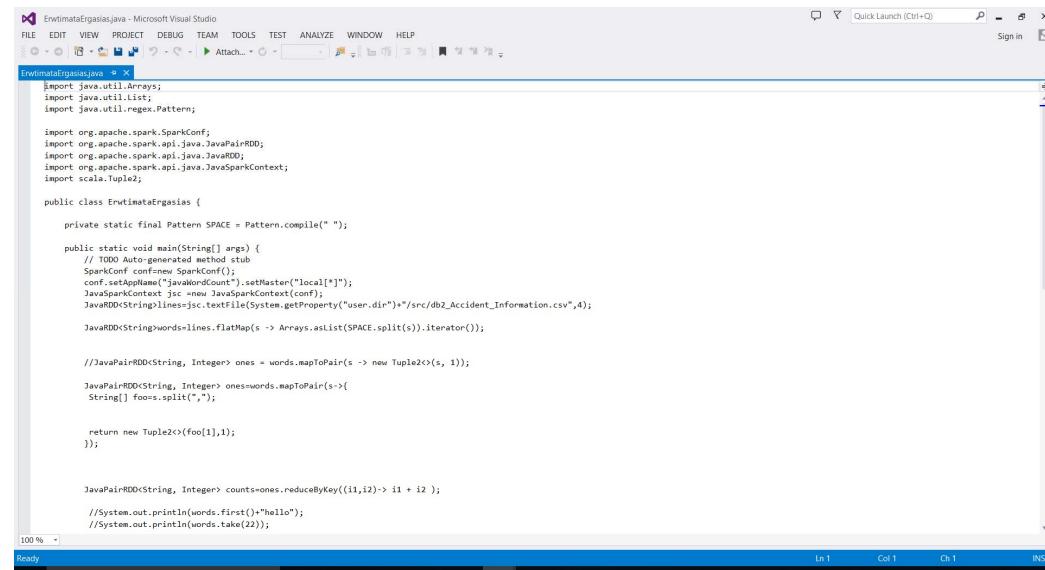
        //JavaPairRDD<String, Integer> ones = words.mapToPair(s-> new Tuple2<s, 1>);

        JavaPairRDD<String, Integer> ones=words.mapToPair(s->{
            String[] foons=split(",");
            return new Tuple2<>(foons[0],1);
        });

        JavaPairRDD<String, Integer> counts=ones.reduceByKey((i1,i2)-> i1 + i2 );

        //System.out.println(words.first()+"hello");
        //System.out.println(words.take(22));
        int x=0;
        List<Tuple2<String, Integer>> output = counts.collect();
        for(Tuple2<,?>tuple : output) {
            System.out.println(tuple._1()+" "+ tuple._2());
            x=x+1;
        }
        System.out.println("How many are the car manufacturers?");
        System.out.println("The car manufacturers are:"+x);
        jsc.close();
    }
}

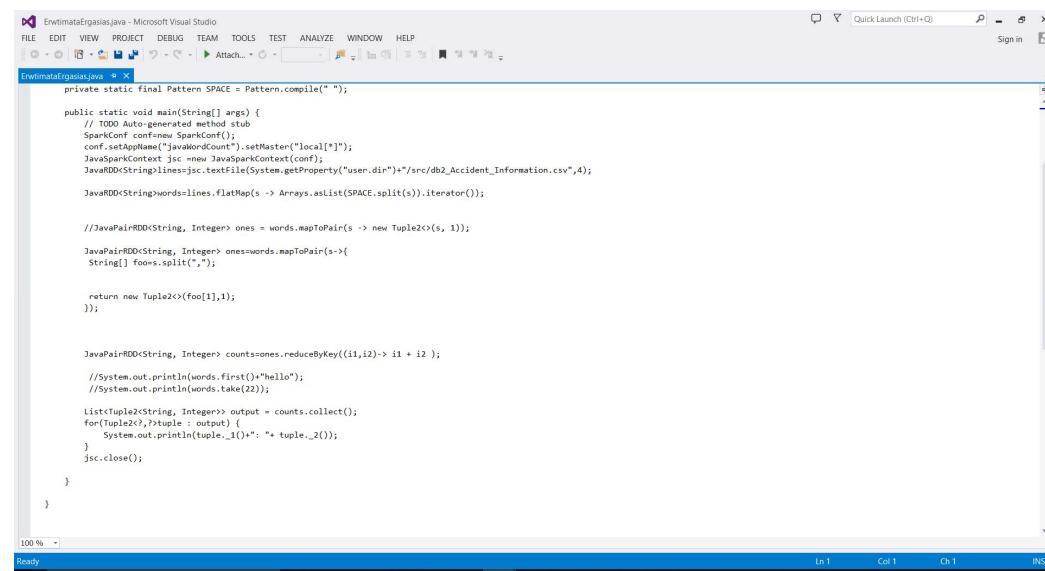
```



```

ErwtimataErgasias.java - Microsoft Visual Studio
FILE EDIT VIEW PROJECT DEBUG TEAM TOOLS TEST ANALYZE WINDOW HELP
Quick Launch (Ctrl+Q) Sign in
ErwtimataErgasias.java + X
private static final Pattern SPACE = Pattern.compile(" ");
public static void main(String[] args) {
    // TODO Auto-generated method stub
    SparkConf conf=new SparkConf();
    conf.setAppName("JavaWordCount");
    conf.setMaster("local[*]");
    JavaSparkContext jsc = new JavaSparkContext(conf);
    JavaRDD<String> lines=jsc.textFile(System.getProperty("user.dir")+"/src/db2_Accident_Information.csv",4);
    JavaRDD<String> words=lines.flatMap(s-> Arrays.asList(SPACE.split(s)).iterator());
    JavaPairRDD<String, Integer> ones = words.mapToPair(s-> new Tuple2<s, 1>());
    JavaPairRDD<String, Integer> counts=ones.reduceByKey((i1,i2)-> i1 + i2 );
    System.out.println(words.first()+"Hello");
    System.out.println(words.take(22));
}

```



```

private static final Pattern SPACE = Pattern.compile(" ");
public static void main(String[] args) {
    // TODO Auto-generated method stub
    SparkConf conf=new SparkConf();
    conf.setAppName("JavaWordCount");
    conf.setMaster("local[*]");
    JavaSparkContext jsc = new JavaSparkContext(conf);
    JavaRDD<String> lines=jsc.textFile(System.getProperty("user.dir")+"/src/db2_Accident_Information.csv",4);
    JavaRDD<String> words=lines.flatMap(s-> Arrays.asList(SPACE.split(s)).iterator());
    JavaPairRDD<String, Integer> ones = words.mapToPair(s-> new Tuple2<s, 1>());
    JavaPairRDD<String, Integer> counts=ones.reduceByKey((i1,i2)-> i1 + i2 );
    System.out.println(words.first()+"Hello");
    System.out.println(words.take(22));
    List<Tuple2<String, Integer>> output = counts.collect();
    for(Tuple2<String, Integer> tuple : output) {
        System.out.println(tuple._1()+" "+ tuple._2());
    }
    jsc.close();
}

```