

Εργασία Βάσεων

ΑΓΓΕΛΙΚΗ ΑΠΟΣΤΟΛΟΥ Π19015

ΑΝΤΩΝΙΑ ΝΙΚΟΛΕΤΤΑ ΓΙΑΝΝΟΥΛΑΚΗ Π19251

Άσκηση 1:

Σχεδιάσαμε το σχεσιακό σχήμα στο Draw.io και βρίσκεται στο relationalSchema.pdf . Οι εντολές Create Table και τα queries βρίσκονται στο αρχείο E and F.sql . Για τη δημιουργία των δεδομένων χρησιμοποιήθηκε το mockaroo. Επίσης στο φάκελο csv βρίσκονται όλα τα αρχεία .csv .

Στη βάση μας έχουμε τους πίνακες **Drivers, Insurance, Cars, InCat, Drivers Address, Violations, Client** και **Violation participants**. Οι πίνακες είναι 3NF για τους εξής λόγους:

Αρχικά είναι 1NF γιατί:

- Όλες οι στήλες έχουν μοναδικό όνομα
- Η σειρά των στοιχείων δεν έχει σημασία
- Όλα τα κελία έχουν μόνο μία τιμή
- Οι τιμές κάθε στήλης είναι του ίδιου τύπου

Είναι 2NF γιατί:

- Είναι 1NF
- Σε κανένα από τους πίνακές μας δεν έχει υπάρξει η ανάγκη να έχουμε παραπάνω από μία σύλη για την ταυτοποίηση μίας

πλειάδας, επομένως τα primary keys μας δεν είναι composite (σύνθετα) και δεν γίνεται να έχουμε Partial Dependency.

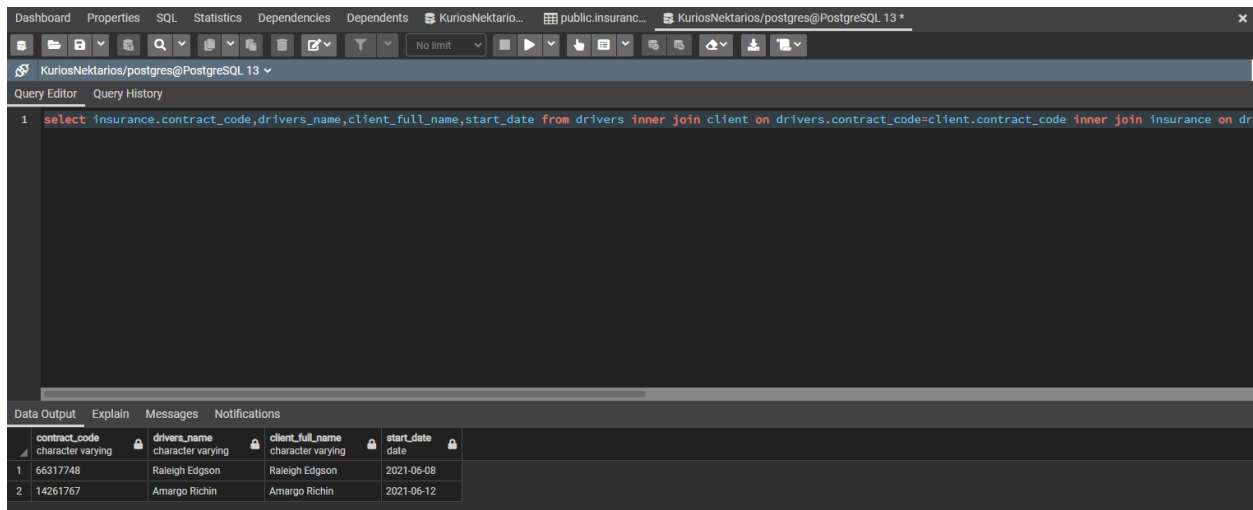
Είναι 3NF γιατί:

- Είναι 2NF
- Κανένα non-prime attribute δεν εξαρτάται από κάποιο άλλο non-prime attribute άρα δεν έχουμε Transitive Dependency.

Άσκηση 2:

Τρέχουμε τα queries του ερωτήματος 2 και παρατηρούμε ότι τα αποτελέσματα είναι τα επιθυμητά:

A)



The screenshot shows a PostgreSQL query editor window. The query editor contains the following SQL query:

```
1 select insurance.contract_code,drivers_name,client_full_name,start_date from drivers inner join client on drivers.contract_code=client.contract_code inner join insurance on dr
```

The results pane shows the following data output:

contract_code	drivers_name	client_full_name	start_date
66317748	Raleigh Edgson	Raleigh Edgson	2021-06-08
14261767	Amargo Richin	Amargo Richin	2021-06-12

B)

The screenshot shows a PostgreSQL query editor with the following SQL query:

```
1 select insurance.contract_code, phone_number,end_date from insurance inner join client on insurance.contract_code=client.contract_code where end_date>date_trunc('month', current_date)
```

The results table is as follows:

contract_code	phone_number	end_date
32021207	653-278-7791	2029-12-01
19100256	231-852-7481	2024-11-28
56720733	666-913-2807	2023-09-25
579560	405-230-4948	2023-11-28
13359359	720-205-2698	2026-03-29
98816141	939-772-5684	2023-04-04
78209711	894-852-8164	2022-11-18

Γ)

The screenshot shows a PostgreSQL query editor with the following SQL query:

```
1 select count (contract_code), insurance_category, case when extract(year from start_date)=2016 then '2016' when extract(year from start_date)=2017 then '2017' when extract(year from start_date) > 2017 then 'Other' end as start_year
```

The results table is as follows:

count	insurance_category	start_year
4	mixed	2016
4	private	2016
4	professional	2016
3	mixed	2017
2	private	2017

Δ1)

The screenshot shows a PostgreSQL query editor with the following query:

```
1 select sum (TO_NUMBER(contract_cost,'L9G999g999.99')) as total ,insurance_category from insurance inner join incat on ins=idincat group by insurance_category order by sum (TO_
```

The results are displayed in a table with the following data:

	total	insurance_category
1	127035	professional
2	166472	private
3	226510	mixed

Δ2)

The screenshot shows a PostgreSQL query editor with the following query:

```
1 select count(contract_code),avg(TO_NUMBER(contract_cost,'L9G999g999.99')) as average_contract_cost,insurance_category, avg(TO_NUMBER(contract_cost,'L9G999g999.99'))*count(cont
```

The results are displayed in a table with the following data:

	count	average_contract_cost	insurance_category	total
1	56	4044.8214285714285714	mixed	999999999984
2	43	2954.3023255813953488	professional	999999999984
3	51	3264.1568627450980392	private	999999999992

E)

The screenshot shows a PostgreSQL query editor with the following SQL query:

```
1 select CASE
2   WHEN EXTRACT(year from current_date)-EXTRACT(year from birthday) <24 then '18-24'
3   WHEN EXTRACT(year from current_date)-EXTRACT(year from birthday) between 25 and 49 then '25-49'
4   WHEN EXTRACT(year from current_date)-EXTRACT(year from birthday) between 50 and 69 then '50-69'
5   else '70+'
6   End as driver_age,count(*)*100/(select count(*)from violation_participants)
7 FROM violation_participants inner join drivers on involved_drivers = drivers_license_number group by driver_age;
```

The Data Output tab shows the following results:

driver_age	?column?
18-24	8
70+	2
50-69	38
25-49	50

Φ)

The screenshot shows a PostgreSQL query editor with the following SQL query:

```
1 select CASE
2   WHEN abs(release_year-EXTRACT(year from current_date)) <5 then '0-5'
3   WHEN abs(release_year-EXTRACT(year from current_date)) between 6 and 10 then '6-10'
4   WHEN abs(release_year-EXTRACT(year from current_date)) between 11 and 15 then '11-15'
5   WHEN abs(release_year-EXTRACT(year from current_date)) between 16 and 20 then '16-20'
6   when abs(release_year-EXTRACT(year from current_date))> 20 then '20+'
7   End as car_age ,count(*)*100/(select count(*)from cars)
8 FROM cars
9 GROUP BY car_age;
```

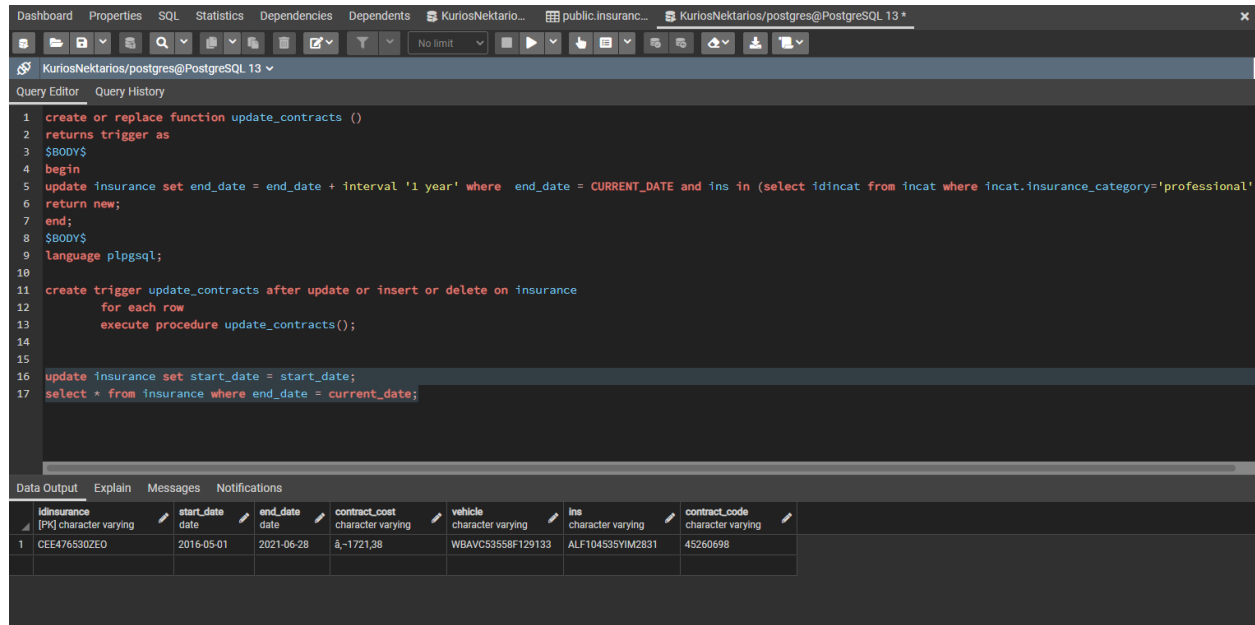
The Data Output tab shows the following results:

car_age	?column?
11-15	23
20+	46
16-20	20
6-10	9

Άσκηση 3:

Τα trigger και cursor βρίσκονται στο αρχείο E and F.sql .

Trigger:



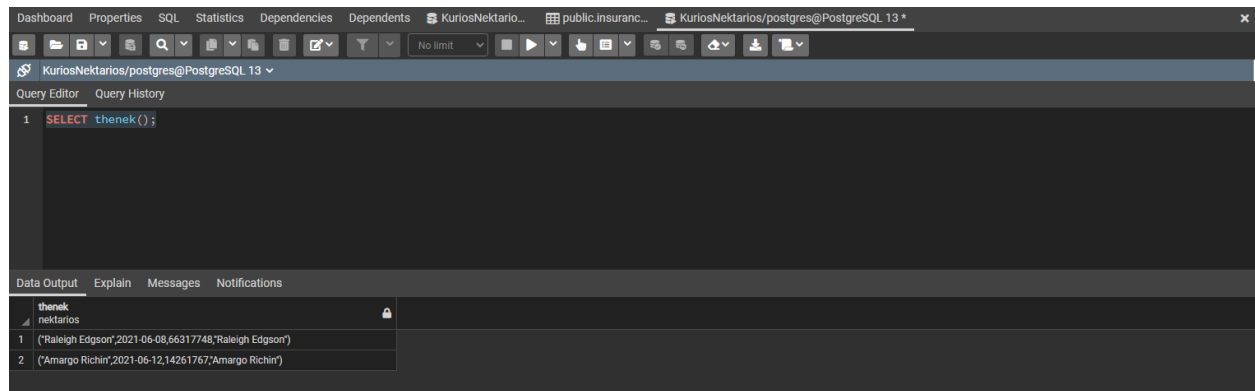
The screenshot shows a PostgreSQL query editor with the following SQL code:

```
1 create or replace function update_contracts ()
2 returns trigger as
3 $BODY$
4 begin
5 update insurance set end_date = end_date + interval '1 year' where end_date = CURRENT_DATE and ins in (select idincat from incat where incat.insurance_category='professional');
6 return new;
7 end;
8 $BODY$
9 language plpgsql;
10
11 create trigger update_contracts after update or insert or delete on insurance
12 for each row
13 execute procedure update_contracts();
14
15
16 update insurance set start_date = start_date;
17 select * from insurance where end_date = current_date;
```

Below the query editor, the "Data Output" tab shows a table with the following data:

idinsurance	start_date	end_date	contract_cost	vehicle	ins	contract_code
CEE476S3OZEO	2016-05-01	2021-06-28	â-1721,38	WBAVC5358F129133	ALF104535YIM2831	45260698

Cursor:



The screenshot shows a PostgreSQL query editor with the following SQL code:

```
1 SELECT thenek();
```

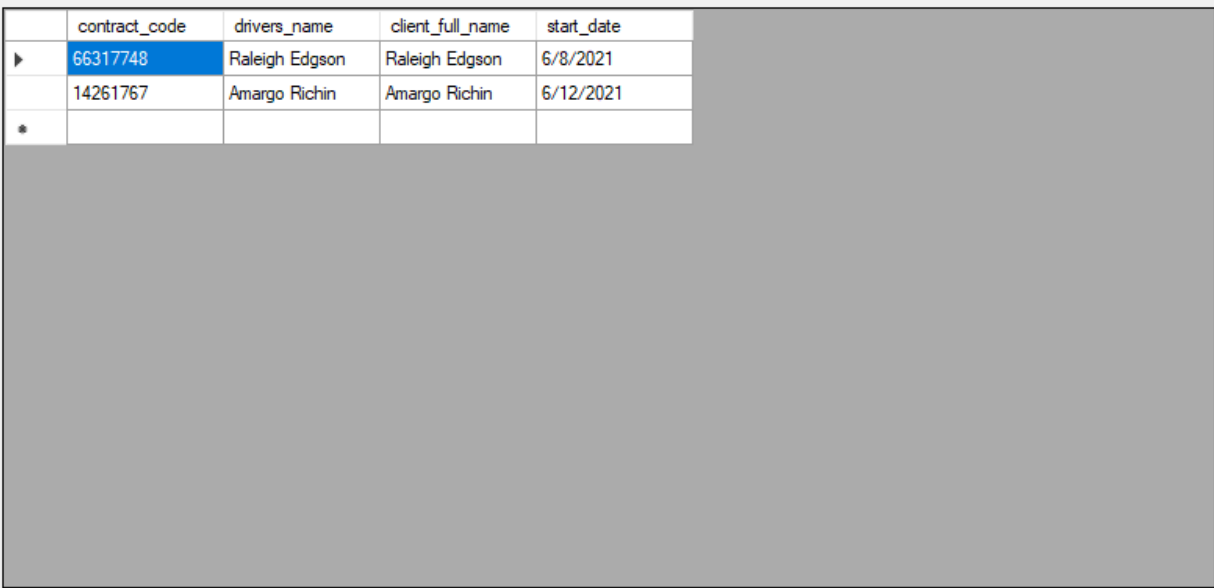
Below the query editor, the "Data Output" tab shows a table with the following data:

thenek
(\"Raleigh Edgson\",2021-06-08,66317748,\"Raleigh Edgson\")
(\"Amargo Richin\",2021-06-12,14261767,\"Amargo Richin\")

Άσκηση 4:

A) select

```
insurance.contract_code,drivers_name,client_full_name,start_date
from drivers inner join client on
drivers.contract_code=client.contract_code inner join insurance on
drivers.contract_code=insurance.contract_code where extract (month
from start_date)=extract (month from current_date) and extract(year
from start_date)= extract(year from current_date );
```



	contract_code	drivers_name	client_full_name	start_date
▶	66317748	Raleigh Edgson	Raleigh Edgson	6/8/2021
	14261767	Amargo Richin	Amargo Richin	6/12/2021
*				

A B C D (1) D (2) E F

B) select insurance.contract_code, phone_number,end_date from insurance inner join client on insurance.contract_code=client.contract_code where end_date>date_trunc('month', current_date + interval '1' month);

Form1

	contract_code	phone_number	end_date
▶	32021207	653-278-7791	12/1/2029
	19100256	231-852-7481	11/28/2024
	56720733	666-913-2807	8/25/2023
	579560	405-230-4948	11/28/2023
	13359359	720-205-2698	3/29/2026
	98816141	939-772-5684	4/4/2023
	78209711	894-852-8164	11/18/2022
	73373529	860-196-3703	10/8/2022
	74665275	982-538-2457	3/10/2025
	9440722	556-240-2490	9/27/2024
	14899959	500-323-5402	9/20/2027
	41633187	809-563-5644	6/5/2023
	50336287	719-664-3323	11/14/2022
	98112213	483-117-6086	8/17/2025
	89761235	461-238-8803	2/17/2024
	63408132	747-968-2588	12/24/2026

A B C D (1) D (2) E F

Γ) select count (contract_code), insurance_category, case when extract(year from start_date)=2016 then '2016' when extract(year from start_date)=2017 then '2017' when extract(year from start_date)=2018 then '2018' when extract(year from start_date)=2019 then '2019' when extract(year from start_date)=2020 then '2020' else 'other' end as start_year from insurance inner join incat on idincat=ins group by insurance_category,start_year order by start_year;

Form1

	count	insurance_category	start_year
▶	4	mixed	2016
	4	private	2016
	4	professional	2016
	3	mixed	2017
	2	private	2017
	4	professional	2017
	7	mixed	2018
	5	private	2018
	4	professional	2018
	3	mixed	2019
	5	private	2019
	4	professional	2019
	10	mixed	2020
	6	private	2020
	5	professional	2020
	29	mixed	other

A B C D (1) D (2) E F

Δ1) select sum (TO_NUMBER(contract_cost,'L9G999g999.99')) as total ,insurance_category from insurance inner join incat on ins=idincat group by insurance_category order by sum

```
(TO_NUMBER(contract_cost,'L9G999g999.99'));
```

	total	insurance_category
▶	127035	professional
	166472	private
	226510	mixed
*		

Δ2) select

```
count(contract_code),avg(TO_NUMBER(contract_cost,'L9G999g999.99'
)) as average_contract_cost,insurance_category,
avg(TO_NUMBER(contract_cost,'L9G999g999.99'))*count(contract_cod
e) as total from insurance inner join incat on ins=idincat group by
```

insurance_category;

Form1

	total	count	insurance_category	average_contract_u
▶	226509.9999999...	56	mixed	4044.821428571...
	127034.9999999...	43	professional	2954.302325581...
	166471.9999999...	51	private	3264.156862745...
*				

A B C D (1) D (2) E F

E)

```
/*E*/
select CASE
  WHEN EXTRACT(year from current_date)-EXTRACT(year from birthday) <24 then '18-24'
  WHEN EXTRACT(year from current_date)-EXTRACT(year from birthday) between 25 and 49 then '25-49'
  WHEN EXTRACT(year from current_date)-EXTRACT(year from birthday) between 50 and 69 then '50-69'
  else '70+'
End as driver_age,count(*)*100/(select count(*)from violation_participants)
FROM violation_participants inner join drivers on involved_drivers = drivers_license_number group by driver_age;
```

Form1

	driver_age	?column?
▶	18-24	8
	70+	2
	50-69	38
	25-49	50
*		

A

B

C

D (1)

D (2)

E

F

Φ)

```
/*F*/
select CASE
    WHEN abs(release_year-EXTRACT(year from current_date)) <5 then '0-5'
    WHEN abs(release_year-EXTRACT(year from current_date)) between 6 and 10 then '6-10'
    WHEN abs(release_year-EXTRACT(year from current_date)) between 11 and 15 then '11-15'
    WHEN abs(release_year-EXTRACT(year from current_date)) between 16 and 20 then '16-20'
    when abs(release_year-EXTRACT(year from current_date))> 20 then '20+'
End as car_age ,count(*)*100/(select count(*)from cars)
FROM cars
GROUP BY car_age;
```

Form1

	car_age	?column?
▶	11-15	23
	20+	46
	16-20	20
	6-10	9
*		

A B C D (1) D (2) E F