

Μάθημα: Βάσεις Δεδομένων

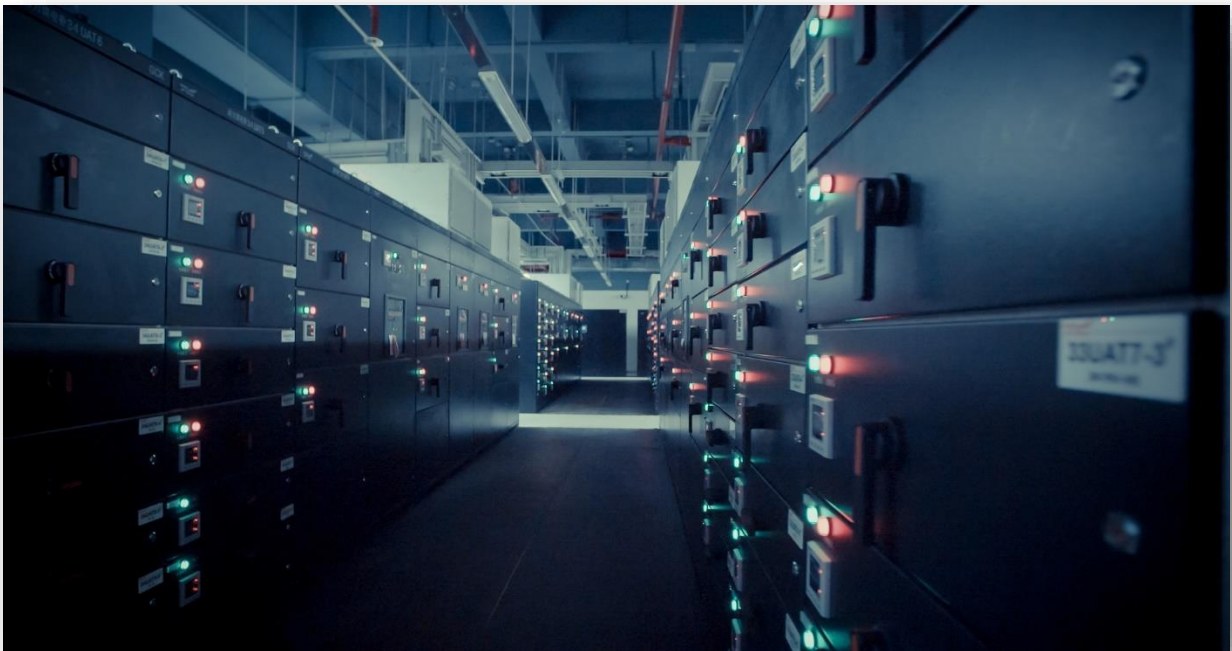
Εργασία 2018 (4^ο εξάμηνο)

Ομάδα Εργασίας

Παναγιώτης Ιωαννίδης (Π16036)

Διονύσης Νίκας (Π16097)

Αθανάσιος Παραβάντης (Π16112)



Περιεχόμενα

1. Εισαγωγή	4
2. Ανάλυση Πινάκων	5
2.1. Car Makes.....	5
2.2. Car Models	6
2.3. Car Warehouse	7
2.4. Customers	8
2.5. Employee	9
2.6. Roles	10
2.7. Sales History	11
2.8. Service History	12
3. Ανάλυση Δομών	13
3.1. Plate Number	13
3.2. Car Condition.....	13
3.3. Sales Action.....	13
4. Θεωρία Κανονικοποίησης	14
5. Επεξήγηση των Queries.....	14
5.1. Μοντέλα αυτοκινήτων με το μέγιστο πλήθος ζημιών.....	14
5.2. Μέσο κέδρος της εταιρίας από επισκευές ανά μήνα	15
5.3. Ο πωλητής με το μέγιστο «τζίρο»	16
5.4. Επισκευές που βρίσκονται σε εκκρεμότητα.....	18
5.5. Εργασίες του τεχνικού 'X' τον τελευταίο μήνα	19
5.6. Αυτοκίνητα που ήρθαν για επισκευή πάνω από 1 φορά τον τελευταίο χρόνο	20
6. Επεξήγηση Trigger & Cursor.....	22
6.1. Trigger	22

6.2. Cursor	25
7. Παράδειγμα σύνδεσης με JDBC	26
7.1. Δομή.....	26
7.2. Εκτέλεση	27

1. Εισαγωγή

Η φετινή εργασία έχει θέμα τη δημιουργία μιας βάσης δεδομένων για εταιρία εμπορίας αυτοκινήτων. Για τις ανάγκες της επιχείρησης, ζητείται η αποθήκευση αρκετών στοιχείων που καθορίζουν σημαντικές πληροφορίες και υποχρεώσεις που πρέπει να εκπληρώσει. Όσο για το τεχνικό κομμάτι, χρησιμοποιήσαμε τη **PostgreSQL** για την δημιουργία της ΒΔ, καθώς και τη γλώσσα προγραμματισμού **Java** ώστε να υλοποιήσουμε μια παραδειγματική εφαρμογή που συνδέεται με τη βάση και αλληλεπιδρά με τα αποθηκευμένα δεδομένα. Η δημιουργία εικονικών καταχωρήσεων έγινε με το εργαλείο **Mockaroo**.

Σύμφωνα με την εκφώνηση της εργασίας, μας ζητούνται δομές και πίνακες που θα υποστηρίζουν την αποθήκευση των παρακάτω στοιχείων της επιχείρησης:

- Στοιχεία πωλητών και τεχνικών
- Ιστορικό επισκευών και σχετικές λεπτομέρειες
- Ιστορικό αγορών και σχετικές λεπτομέρειες
- Αυτοκίνητα που διαχειρίζεται η εταιρία
- Ιδιώτες που συναλλάσσονται με την εταιρία

Στην αμέσως επόμενη ενότητα θα αναλύσουμε τη σχεδίαση της βάσης δεδομένων για την υποστήριξη των παραπάνω πληροφοριών.

2. Ανάλυση Πινάκων

Όλες οι εντολές SQL για τη δημιουργία των παρακάτω πινάκων βρίσκονται στο αρχείο **part1/all.sql** και ξεχωριστά η καθεμία στο φάκελο **part1**. Χρησιμοποιώντας το query tool του pgAdmin, εκτελώντας ότι βρίσκεται στο part1/all.sql θα δημιουργηθεί επιτυχώς η βάση δεδομένων της εργασίας.

2.1. Car Makes

Ο πίνακας car_makes αποθηκεύει κωδικούς εταιριών κατασκευής αυτοκινήτων και τις αντίστοιχες ονομασίες τους.

Χαρακτηριστικά:

- **id** (20 characters)
- **title** (20 characters)

Πρωτεύον Κλειδί:

- **id**: Κάθε εταιρία αντιπροσωπεύεται από έναν μοναδικό κωδικό.

Ξένα Κλειδιά:

- Κανένα

```
CREATE TABLE car_makes (  
  id VARCHAR(20) NOT NULL,  
  title VARCHAR(60) NOT NULL,  
  
  PRIMARY KEY (id)  
);
```

2.2. Car Models

Ο πίνακας `car_models` αποθηκεύει κωδικούς μοντέλων αυτοκινήτων, την εταιρία κατασκευαστή και την ονομασία του μοντέλου.

Χαρακτηριστικά

- **make_id** (20 characters)
- **id** (20 characters)
- **title** (60 characters)

Πρωτεύον Κλειδί

- **id**: Κάθε μοντέλο αμαξιού αντιπροσωπεύεται από έναν μοναδικό κωδικό.

Ξένα Κλειδιά

- **make_id**: αναφορά στο **id** του πίνακα **car_makes**.

```
CREATE TABLE car_models (  
  make_id VARCHAR(20) NOT NULL REFERENCES car_makes(id),  
  id VARCHAR(20) NOT NULL ,  
  title VARCHAR(60) NOT NULL ,  
  
  PRIMARY KEY (id)  
);
```

2.3. Car Warehouse

Ο πίνακας `car_warehouse` αποθηκεύει τα αυτοκίνητα που διαχειρίζεται η εταιρία.

Χαρακτηριστικά:

- **id** (serial)
- **owner_id** (numeric 8, 0)
- **plate** (plate_num, unique)
- **model_id** (20 characters)
- **manufacturing_date** (numeric 4, 0)
- **condition** (car_condition)

Πρωτεύον Κλειδί:

- **id**: Κάθε εγγραφή χαρακτηρίζεται από έναν αύξον αριθμό.

Ξένα Κλειδιά:

- **owner_id** αναφορά στο **afm** του πίνακα **customers**.
- **model_id** αναφορά στο **id** του πίνακα **car_models**.

```
CREATE TABLE car_warehouse (  
  id SERIAL NOT NULL ,  
  owner_id NUMERIC(8,0) REFERENCES customers(afm),  
  plate PLATE_NUM UNIQUE,  
  model_id VARCHAR(20) NOT NULL REFERENCES car_models(id),  
  manufacturing_date NUMERIC(4,0),  
  condition CAR_CONDITION NOT NULL,  
  
  PRIMARY KEY (id)  
);
```

2.4. Customers

Ο πίνακας customers αποθηκεύει τα στοιχεία πελατών που συναλλάσσονται με την εταιρία.

Χαρακτηριστικά:

- **afm** (numeric 8, 0)
- **first_name** (60 characters)
- **last_name** (60 characters)
- **email** (60 characters)
- **phone** (60 characters)

Πρωτεύον Κλειδί:

- **afm:** Κάθε πελάτης έχει ένα μοναδικό αριθμό ΑΦΜ.

Ξένα Κλειδιά:

- Κανένα

```
CREATE TABLE customers (  
  afm NUMERIC(8,0) NOT NULL,  
  first_name VARCHAR(60) NOT NULL ,  
  last_name VARCHAR(60) NOT NULL ,  
  email VARCHAR(60) NOT NULL UNIQUE ,  
  phone VARCHAR(10) NOT NULL UNIQUE ,  
  
  PRIMARY KEY (afm)  
);
```


2.5. Employee

Ο πίνακας employee αποθηκεύει τους εργαζόμενους της εταιρίας που χωρίζονται σε δυο κατηγορίες: τεχνικοί και πωλητές.

Χαρακτηριστικά:

- **afm** (numeric 8, 0)
- **role_id** (4 characters)
- **first_name** (60 characters)
- **last_name** (60 characters)
- **email** (60 characters)

Πρωτεύον Κλειδί:

- **afm**: Κάθε εργαζόμενος έχει ένα μοναδικό αριθμό ΑΦΜ.

Ξένα Κλειδιά:

- **role_id** αναφέρεται στο **id** του πίνακα **roles**.

```
CREATE TABLE employee (  
  afm NUMERIC(8,0) NOT NULL,  
  role_id VARCHAR(4) REFERENCES roles(id),  
  first_name VARCHAR(60) NOT NULL ,  
  last_name VARCHAR(60) NOT NULL ,  
  email VARCHAR(60) NOT NULL UNIQUE,  
  
  PRIMARY KEY (afm)  
);
```

2.6. Roles

Ο πίνακας roles αποθηκεύει όλους τους πιθανούς ρόλους των εργαζομένων της εταιρίας.

Χαρακτηριστικά:

- **id** (4 characters)
- **title** (40 characters)

Πρωτεύον Κλειδί:

- **id**: Κάθε τίτλος έχει έναν μοναδικό κωδικό id.

Ξένα Κλειδιά:

- Κανένα

```
CREATE TABLE roles (  
  id VARCHAR(4) NOT NULL ,  
  title VARCHAR(40) NOT NULL ,  
  
  PRIMARY KEY (id)  
);
```

2.7. Sales History

Ο πίνακας sales_history αποθηκεύει το ιστορικό αγορών και πωλήσεων της εταιρίας.

Χαρακτηριστικά:

- **id** (serial)
- **salesman_id** (numeric 8, 0)
- **car_warehouse_id** (integer)
- **price** (float)
- **action** (sales_action)
- **date** (date)

Πρωτεύον Κλειδί:

- **id**: Κάθε εγγραφή χαρακτηρίζεται από έναν αύξον αριθμό.

Ξένα Κλειδιά:

- **salesman_id**: αναφέρεται στο **afm** του πίνακα **employees**.
- **car_warehouse_id**: αναφέρεται στο **id** του πίνακα **car_warehouse**.

```
CREATE TABLE sales_history (  
  id serial NOT NULL ,  
  salesman_id NUMERIC(8,0) NOT NULL REFERENCES employee(afm),  
  car_warehouse_id INT NOT NULL REFERENCES car_warehouse(id),  
  price FLOAT NOT NULL ,  
  action SALES_ACTION NOT NULL ,  
  date DATE,  
  
  PRIMARY KEY (id)  
);
```

2.8. Service History

Ο πίνακας service_history αποθηκεύει το ιστορικό του service της εταιρίας.

Χαρακτηριστικά:

- **id** (serial)
- **car_warehouse_id** (integer)
- **tech_id** (numeric 8, 0)
- **cost** (float)
- **start_date** (date)
- **end_date** (date)

Πρωτεύον Κλειδί:

- **id**: Κάθε εγγραφή χαρακτηρίζεται από έναν αύξον αριθμό.

Ξένα Κλειδιά:

- **tech_id**: αναφορά στο **afm** του πίνακα **employee**.
- **car_warehouse_id**: αναφορά στο **id** του πίνακα **car_warehouse**.

```
CREATE TABLE service_history (  
    id SERIAL NOT NULL ,  
    car_warehouse_id INT NOT NULL REFERENCES car_warehouse(id),  
    tech_id NUMERIC(8,0) NOT NULL REFERENCES employee(afm),  
    cost FLOAT NOT NULL ,  
    start_date DATE NOT NULL ,  
    end_date DATE,  
  
    PRIMARY KEY (id)  
);
```

3. Ανάλυση Δομών

Όλες οι εντολές SQL για τη δημιουργία των παρακάτω δομών βρίσκονται στο αρχείο **part1/all.sql** και ξεχωριστά η καθεμία στο φάκελο **part1**.

3.1. Plate Number

Χρησιμοποιείται για την αναπαράσταση πινάκων κυκλοφορίας.

- **plate_char** (characters)
- **plate_number** (numeric)

```
CREATE TYPE PLATE_NUM AS (  
    plate_char CHAR(3),  
    plate_number NUMERIC(4,0));
```

3.2. Car Condition

Χρησιμοποιείται για την κατάσταση του αυτοκινήτου.

- Enum: new, used

```
CREATE TYPE CAR_CONDITION AS ENUM ('new', 'used');
```

3.3. Sales Action

Χρησιμοποιείται για τον διαχωρισμό των αγορών από τις πωλήσεις.

- Enum: buy, sale

```
CREATE TYPE SALES_ACTION AS ENUM ('buy', 'sale');
```

4. Θεωρία Κανονικοποίησης

Η εφαρμογή της θεωρίας κανονικοποίησης πάνω στο σχεσιακό σχήμα της ΒΔ βρίσκεται στο αρχείο **bcnf_3nf.txt**.

5. Επεξήγηση των Queries

Όλα τα queries για τα ερωτήματα που ζητούνται στην εργασία βρίσκονται στο αρχείο **part2/all.sql** και ξεχωριστά το καθένα στο φάκελο **part2**.

5.1. Μοντέλα αυτοκινήτων με το μέγιστο πλήθος ζημιών

Για την απλούστευση του ερωτήματος, δημιουργούμε ένα view το οποίο περιέχει όλα τα μοντέλα αμαξιών και πόσες φορές έχει έρθει το καθένα για service, ανεξάρτητα αν πολλοί διαφορετικοί πελάτες έχουν το ίδιο αμάξι.

```
CREATE VIEW COUNT_MODELS AS (  
  SELECT  
    car_models.title,  
    COUNT(car_models.id) AS num_of_model  
  FROM  
    car_warehouse  
    INNER JOIN car_models ON car_warehouse.model_id = car_models.id  
  GROUP BY  
    car_models.title  
);  
  
SELECT  
  *  
FROM  
  COUNT_MODELS  
WHERE  
  COUNT_MODELS.num_of_model = (SELECT max(num_of_model) AS maximum FROM COUNT_MODELS);
```

Ενδεικτικά αποτελέσματα του query:

Data Output			Explain	Messages	Query History
	title		num_of_model		
	character varying (60)		bigint		
1	Vanquish		56		

5.2. Μέσο κέδρος της εταιρίας από επισκευές ανά μήνα

Το συγκεκριμένο ερώτημα είναι αρκετά απλό και δεν χρειάζεται χρήση view για την απλούστευση του. Γίνεται στρογγυλοποίηση του κέρδους για τη προβολή του.

```
SELECT
  to_char(start_date, 'Month') as month,
  ROUND(AVG(cost)::numeric, 2) as avg_profit
FROM
  service_history
GROUP BY
  to_char(start_date, 'Month')
ORDER BY
  avg_profit DESC
```

Ενδεικτικά αποτελέσματα του query:

	Data Output	Explain	Messages	Query History
	month text	avg_profit numeric		
1	May	538.57		
2	June	534.78		
3	December	528.80		
4	February	526.68		
5	January	516.63		
6	November	492.88		
7	September	482.37		
8	April	473.58		
9	August	465.56		
10	July	460.68		
11	October	450.01		
12	March	444.18		

5.3. Ο πωλητής με το μέγιστο «τζίρο»

Αυτό το ερώτημα γίνεται αρκετά απλούστερο με τη χρήση views όπως βλέπουμε παρακάτω.

Δημιουργήσαμε τρία views για τις πωλήσεις, τις αγορές και το κέρδος:

```
CREATE VIEW SALES AS (  
  SELECT  
    A.salesman_id,  
    SUM(A.price) AS sale  
  FROM  
    sales_history AS A  
  WHERE  
    A.action = 'sale'  
  GROUP BY  
    A.salesman_id  
);  
  
DROP VIEW IF EXISTS BUYS;  
CREATE VIEW BUYS AS (  
  SELECT  
    A.salesman_id,  
    SUM(A.price) AS buy  
  FROM  
    sales_history AS A  
  WHERE  
    A.action = 'buy'  
  GROUP BY  
    A.salesman_id  
);  
  
DROP VIEW IF EXISTS PROFIT;  
CREATE VIEW PROFIT AS (  
  SELECT  
    SALES.salesman_id,  
    (SALES.sale - BUYS.buy) AS profit  
  FROM  
    SALES  
    NATURAL JOIN BUYS  
);  
  
SELECT  
  salesman_id,  
  employee.first_name,  
  employee.last_name,  
  profit  
FROM  
  PROFIT  
  INNER JOIN employee ON PROFIT.salesman_id = employee.afm  
WHERE  
  profit = (SELECT MAX(profit) FROM PROFIT);
```


Ενδεικτικά αποτελέσματα του query:

Data Output Explain Messages Query History				
	salesman_id numeric (8)	first_name character varying (60)	last_name character varying (60)	profit double precision
1	17983590	Garrek	Nevinson	2430997.22

5.4. Επισκευές που βρίσκονται σε εκκρεμότητα

Πολύ απλό ερώτημα στο οποίο ο μόνος έλεγχος είναι στην ύπαρξη της τελικής ημερομηνίας.

```
SELECT
  *
FROM
  service_history
WHERE
  end_date IS NULL
```

Ενδεικτικά αποτελέσματα του query:

Data Output	Explain	Messages	Query History				
	id integer	car_warehouse_id integer	tech_id numeric (8)	cost double precision	start_date date	end_date date	
1	8	498	85601262	739.14	2018-05-21	[null]	
2	18	160	85601262	437.6	2018-05-22	[null]	
3	28	387	85601262	691.38	2018-05-25	[null]	
4	30	118	85601262	129.52	2018-05-28	[null]	
5	44	179	85601262	908.98	2018-05-30	[null]	
6	100	428	55241214	317.22	2018-05-29	[null]	
7	141	496	55241214	642	2018-05-26	[null]	
8	312	387	29928702	991.55	2018-05-28	[null]	
9	367	427	29928702	362.61	2018-05-30	[null]	
10	376	459	29928702	707.37	2018-05-24	[null]	
11	420	424	55241214	851.6	2018-05-27	[null]	
12	425	355	55241214	954.74	2018-05-28	[null]	
13	448	404	41235675	637.83	2018-05-29	[null]	

5.5. Εργασίες του τεχνικού 'X' τον τελευταίο μήνα

Για τις ανάγκες του ερωτήματος επιλέξαμε τον τεχνικό με ΑΦΜ 85601262.

```
SELECT
    first_name,
    last_name,
    car_warehouse_id,
    start_date,
    end_date
FROM
    employee
    INNER JOIN service_history ON employee.afm = service_history.tech_id
WHERE
    employee.afm = 85601262
    AND (end_date > NOW() - INTERVAL '1 month' OR (end_date IS NULL AND start_date > NOW() - INTERVAL '1 month'))
```

Ενδεικτικά αποτελέσματα του query:

	first_name character varying (60)	last_name character varying (60)	car_warehouse_id integer	start_date date	end_date date
1	Livy	Collaton	498	2018-05-21	[null]
2	Livy	Collaton	160	2018-05-22	[null]
3	Livy	Collaton	387	2018-05-25	[null]
4	Livy	Collaton	118	2018-05-28	[null]
5	Livy	Collaton	179	2018-05-30	[null]

5.6. Αυτοκίνητα που ήρθαν για επισκευή πάνω από 1 φορά τον τελευταίο χρόνο

Σε αυτό το ερώτημα έχουμε κάνει αρκετά join, groups και ordering για τη σωστή απάντηση με τα κατάλληλα στοιχεία. Γίνεται έλεγχος και για την ημερομηνία αρχής και τέλους των επισκευών.

```
SELECT
    car_models.title,
    car_warehouse.plate,
    customers.first_name,
    customers.last_name,
    COUNT(car_models.id) AS service_count
FROM
    service_history
    INNER JOIN car_warehouse ON service_history.car_warehouse_id = car_warehouse.id
    INNER JOIN customers ON car_warehouse.owner_id = customers.afm
    INNER JOIN car_models ON car_warehouse.model_id = car_models.id
WHERE
    service_history.start_date <= date_trunc('year', now()) - interval '1 year'
    OR
    service_history.end_date <= date_trunc('year', now()) - interval '1 year'
GROUP BY
    car_models.id,
    car_warehouse.plate,
    customers.afm
HAVING
    COUNT(car_models.id) > 1
ORDER BY
    service_count DESC
```

Ενδεικτικά αποτελέσματα του query:

Data Output Explain Messages Query History					
	title character varying (60)	plate plate_num	first_name character varying (60)	last_name character varying (60)	service_count bigint
1	NSX	(CRN,5188)	Dunstan	Josovitz	6
2	- ES 250	(SCI,8118)	Shelli	Briereton	5
3	Eagle	(XIM,2533)	Kent	Iohananof	5
4	Quattro	(DCY,6257)	Meredith	Behan	4
5	4000	(MJQ,9342)	Winfred	Trevers	4
6	allroad	(YYE,6740)	Cookie	Saleway	4
7	CL Models (4)	(YSR,4400)	Rodina	Ghidoni	3
8	RDX	(TRX,2756)	Mario	Fursey	3
9	RDX	(XWV,9116)	Shannen	Karlowicz	3
10	RL Models (2)	(MRQ,3763)	Damon	Hagyard	3
11	RS 4	(DXE,1904)	Tarah	Boteman	3
12	Vanquish	(XZR,1880)	Cristine	Sibson	3
13	- 2.2CL	(OON,4321)	Camala	Oswal	3
14	NSX	(TYV,5149)	Hollie	Test	3
15	Legend	(WGW,2883)	Cristine	Sibson	3

6. Επεξήγηση Trigger & Cursor

Ο trigger και cursor που ζητούνται στην εργασία βρίσκονται στο αρχείο **part3/all.sql** και ξεχωριστά το καθένα στο φάκελο **part3**.

6.1. Trigger

Σε αυτό ερώτημα κατασκευάσαμε έναν trigger ο οποίος υλοποιεί την αυτόματη ενημέρωση του ιστορικού των αυτοκινήτων που διατηρείται από την εταιρεία.

Η υλοποίηση που επιλέξαμε να πραγματοποιήσουμε είναι όσο το δυνατόν πιο κοντά στη λειτουργία μια πραγματικής εφαρμογής και καλύπτει τις, πιθανές, δυο περιπτώσεις:

1. Να υπάρχει ήδη στο ιστορικό των αυτοκινήτων (car_warehouse) το αυτοκίνητο που θα εισαχθεί στο service_history.
2. Να μην υπάρχει στο ιστορικό των αυτοκινήτων (car_warehouse) το αυτοκίνητο που θα εισαχθεί στο service_history. Εκεί θα πρέπει πρώτα να δημιουργήσουμε την εγγραφή με τα στοιχεία του νέου πελάτη (σχέση customers), δεύτερον θα δημιουργήσουμε την εγγραφή με τα στοιχεία του αυτοκινήτου (σχέση car_warehouse) και τέλος τα απαραίτητα στοιχεία του αυτοκινήτου που ήρθε στον πίνακα των επισκευών (σχέση service_history).

Για να πετύχουμε την αλυσιδωτή υλοποίηση που έχει περιγραφεί στην δεύτερη περίπτωση, δημιουργούμε ένα προσωρινό πίνακα με όνομα temp_car_service οπου περιέχει τα καταλληλά χαρακτηριστικά.

Στη συνέχεια, δημιουργούμε τη συνάρτηση που θα εκτελείται από τον trigger. Αρχικά γίνεται έλεγχος για τον αν υπάρχει ο αριθμός κυκλοφορίας στον πίνακα με τα στοιχεία των αυτοκινήτων (car_warehouse). Αν υπάρχει τότε εκτελείται η πρώτη περίπτωση, δηλαδή η απευθείας εισαγωγή των στοιχείων στον πίνακα service_history.

Αν δεν υπάρχει ο αριθμός κυκλοφορίας τότε:

1. Γίνεται η εισαγωγή των στοιχείων του πελάτη στον πίνακα customers.
2. Εισάγονται τα στοιχεία του αυτοκινήτου στον πίνακα car_warehouse.
3. Τέλος, εισάγονται τα στοιχεία στον πίνακα service history.

Αμέσως μετά δημιουργούμε τον trigger πάνω στον προσωρινό πίνακα temp_car_service. Ορίζουμε ότι ο trigger θα πρέπει να ενεργοποιείται πριν εισαγωγή των στοιχείων σε αυτόν τον πίνακα.

Ας δούμε ένα παράδειγμα στο οποίο εισάγουμε τα εξής στον πίνακα temp_car_service:

```
INSERT INTO temp_car_service(first_name, last_name, afm, email, phone,
                             plate, model_id, manufacturing_date,
                             tech_id, cost, start_date, end_date)
VALUES ('Giannis', 'Kontoulis', 99112233, 'ikontoulis@unipi.gr', '2101425859',
        ROW('ZXM', '2172'), 'SCIR', 2011,
        85601262, 150.99, now(), null);
```

Έχοντας δημιουργήσει το αντίστοιχο function και trigger, παρατηρούμε ότι ο πελάτης έχει καταχωρηθεί στους customers:

1select * from customers where afm=99112233

Data Output

Explain

Messages

Query History

	afm numeric (8)	first_name character varying (60)	last_name character varying (60)	email character varying (60)	phone character varying (10)
1	99112233	Giannis	Kontoulis	ikontoulis@unipi.gr	2101425859

Το αμάξι έχει καταχωρηθεί στο car_warehouse:

1

```
select * from car_warehouse where owner_id=99112233;
```

Data Output

[Explain](#)

[Messages](#)

[Query History](#)

	id integer	owner_id numeric (8)	plate plate_num	model_id character varying (20)	manufacturing_date numeric (4)	condition car_condition
1	500	99112233	(ZXM,2172)	SCIR	2011	used

Και τέλος, η επισκευή έχει καταχωρηθεί στο service_history:

1

select * from service_history where car_warehouse_id=500

Data Output

Explain

Messages

Query History

	id integer	car_warehouse_id integer	tech_id numeric (8)	cost double precision	start_date date	end_date date
1	508	500	85601262	150.99	2018-05-31	[null]

6.2. Cursor

Για τον cursor που μας ζητείται, υλοποιήσαμε το 6^ο query της εργασίας. Όπως και στη προηγούμενη ενότητα για την επεξήγησή του, εμφανίζονται τα ίδια αποτελέσματα, δηλαδή αυτοκίνητα που ήρθαν για επισκευή πάνω από 1 φορά τον τελευταίο χρόνο.

Ουσιαστικά, φτιάχνουμε ένα temp_type που θα χρησιμοποιηθεί στον cursor για να επιστρέψει ένα setof temp_type. Με αυτό το τρόπο μπορούμε εύκολα να επιστρέψουμε τα ίδια αποτελέσματα όπως στο 6^ο query, με τη χρήση του cursor.

Data Output							Explain	Messages	Query History
	title character (60)		plate plate_num	first_name character (60)	last_name character (60)	service_count bigint			
1	NSX	...	(CRN,5188)	Dunstan	...	Josovitz	...	6	
2	- ES 250	...	(SCI,8118)	Shelli	...	Briereton	...	5	
3	Eagle	...	(XIM,2533)	Kent	...	Iohananof	...	5	
4	Quattro	...	(DCY,6257)	Meredith	...	Behan	...	4	
5	4000	...	(MJQ,9342)	Winfred	...	Trevers	...	4	
6	allroad	...	(YYE,6740)	Cookie	...	Saleway	...	4	
7	CL Models (4)	...	(YSR,4400)	Rodina	...	Ghidoni	...	3	
8	RDX	...	(TRX,2756)	Mario	...	Fursey	...	3	
9	RDX	...	(XWV,9116)	Shannen	...	Karlowicz	...	3	
10	RL Models (2)	...	(MRQ,3763)	Damon	...	Hagyard	...	3	
11	RS 4	...	(DXE,1904)	Tarah	...	Boteman	...	3	
12	Vanquish	...	(XZR,1880)	Cristine	...	Sibson	...	3	
13	- 2.2CL	...	(OON,4321)	Camala	...	Oswal	...	3	
14	NSX	...	(TYV,5149)	Hollie	...	Test	...	3	
15	Legend	...	(WGW,2883)	Cristine	...	Sibson	...	3	
16	Legend	...	(GUV,4860)	Leia	...	Klempke	...	3	
17	TL Models (3)	...	(EMJ,6536)	Wenonah	...	Leman	...	3	
18	Spirit	...	(IPE,2281)	Anjela	...	Vidineev	...	3	
19	DB7	...	(MIB,4753)	Meredith	...	Behan	...	2	

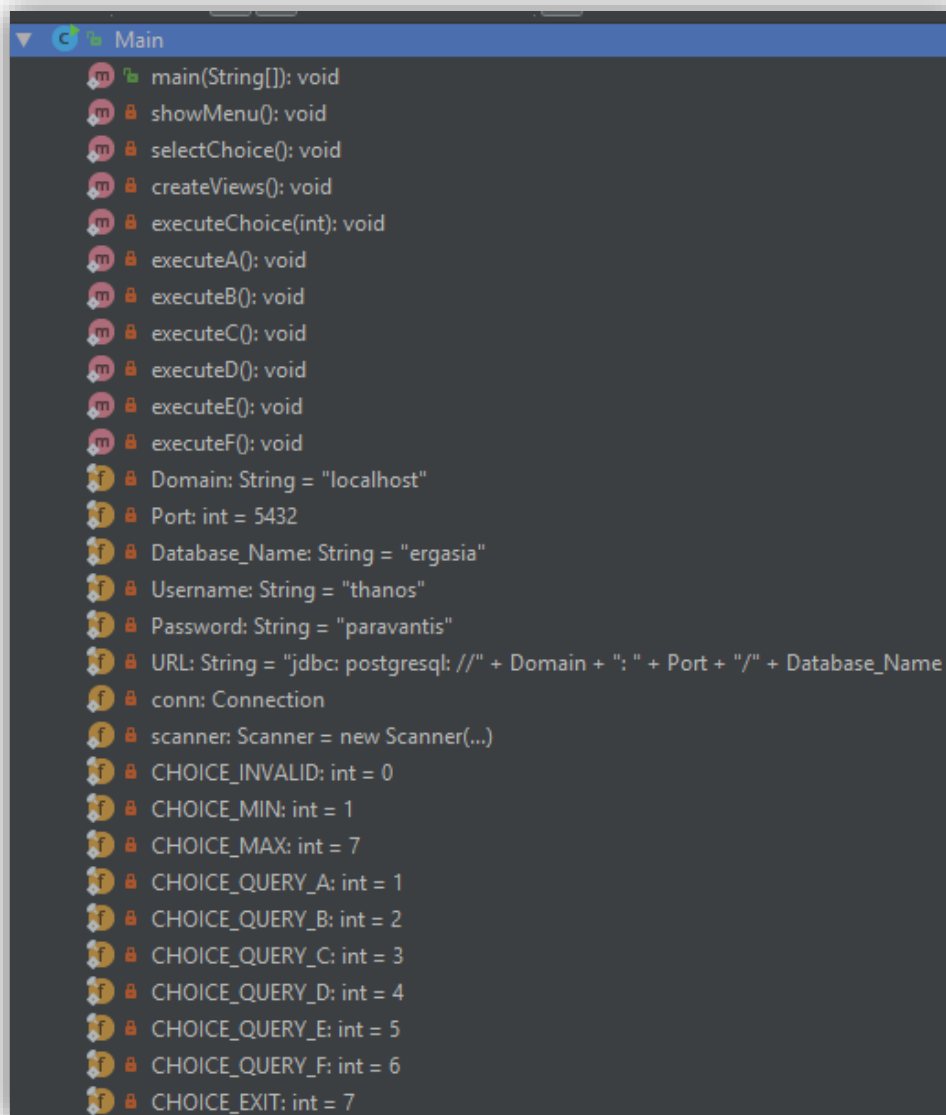
7. Παράδειγμα σύνδεσης με JDBC

Το project βρίσκεται στον φάκελο **JDBC_Example**.

7.1. Δομή

Σύμφωνα με την εκφώνηση της άσκησης, υλοποιήσαμε σε Java ένα πρόγραμμα που χρησιμοποιεί το JDBC ώστε να συνδεθεί με τη βάση δεδομένων και να εκτελέσει ερωτήματα.

Αποτελείται από μόνο ένα αρχείο Main.java όπου έχει τις εξής μεθόδους και ιδιότητες:



```
▼ Main
main(String[]): void
showMenu(): void
selectChoice(): void
createViews(): void
executeChoice(int): void
executeA(): void
executeB(): void
executeC(): void
executeD(): void
executeE(): void
executeF(): void
Domain: String = "localhost"
Port: int = 5432
Database_Name: String = "ergasia"
Username: String = "thanos"
Password: String = "paravantis"
URL: String = "jdbc: postgresql: //" + Domain + ": " + Port + "/" + Database_Name
conn: Connection
scanner: Scanner = new Scanner(...)
CHOICE_INVALID: int = 0
CHOICE_MIN: int = 1
CHOICE_MAX: int = 7
CHOICE_QUERY_A: int = 1
CHOICE_QUERY_B: int = 2
CHOICE_QUERY_C: int = 3
CHOICE_QUERY_D: int = 4
CHOICE_QUERY_E: int = 5
CHOICE_QUERY_F: int = 6
CHOICE_EXIT: int = 7
```

Η εφαρμογή εκκινεί με τη κλήση της `main()` και στη συνέχεια η μέθοδος `showMenu()` καλείται για να εμφανίσει ένα μενού επιλογών. Κάθε φορά που γίνεται μια επιλογή από το χρήστη μέσω της κονσόλας, εκτελείται η κατάλληλη μέθοδος για να καλύψει το ερώτημα.

7.2. Εκτέλεση

Με την εκτέλεση του προγράμματος εμφανίζεται το μενού επιλογών:

```
Select which query to execute:
1. Erotima A
2. Erotima B
3. Erotima C
4. Erotima D
5. Erotima E
6. Erotima F
7. Exit
>
```

Ο χρήστης πρέπει να εισάγει ένα νούμερο από το 1 μέχρι το 7 για να εκτελεστεί η αντίστοιχη λειτουργία. Αφού εμφανιστούν τα αντίστοιχα αποτελέσματα, το μενού ξαναεμφανίζεται. Σημειώνεται ότι αν γίνει λάθος εισαγωγή αριθμού, εμφανίζονται αντίστοιχα μηνύματα λάθους.

```
Select which query to execute:
1. Erotima A
2. Erotima B
3. Erotima C
4. Erotima D
5. Erotima E
6. Erotima F
7. Exit
> 0
Please enter a valid number from 1 to 7.
> -1
Please enter a valid number from 1 to 7.
> !!!!!!!!!!!!!!!!!!!!!!!!!!!!!
Please enter a valid number from 1 to 7.
>
```

Αποτελέσματα εκτέλεση του ερωτήματος A:

Erotima A	
Title	Number of Models
Vanquish	56

Αποτελέσματα εκτέλεσης του ερωτήματος B:

```
Erotima B
Month      Average Profit
May        538.57
June       534.78
December   528.80
February   526.68
January    516.63
November   492.88
September  482.37
April      473.58
August     465.56
July       460.68
October    450.01
March      444.18
```

Αποτελέσματα εκτέλεσης του ερωτήματος C:

```
Erotima C
Salesman ID      First Name      Last Name      Profit
17983590         Garrek         Nevinson       2430997
```

Αποτελέσματα εκτέλεσης του ερωτήματος D:

```
 Erotima D
```

ID	Car Warehouse ID	Tech ID	Cost	Start Date	End Date
8	498	85601262	739	2018-05-21	null
18	160	85601262	437	2018-05-22	null
28	387	85601262	691	2018-05-25	null
30	118	85601262	129	2018-05-28	null
44	179	85601262	908	2018-05-30	null
100	428	55241214	317	2018-05-29	null
141	496	55241214	642	2018-05-26	null
312	387	29928702	991	2018-05-28	null
367	427	29928702	362	2018-05-30	null
376	459	29928702	707	2018-05-24	null
420	424	55241214	851	2018-05-27	null
425	355	55241214	954	2018-05-28	null
448	404	41235675	637	2018-05-29	null

Στο ερώτημα E απαραίτητη προϋπόθεση είναι η εισαγωγή του ΑΦΜ για την εμφάνιση σχετικών εργασιών του τεχνικού. Εάν ο χρήστης πληκτρολογήσει χαρακτήρες τότε το ΑΦΜ 85601262 θα χρησιμοποιηθεί αυτόματα. Διαφορετικά, αν δεν υπάρχει τεχνικός με το επιλεγμένο ΑΦΜ τότε δεν εμφανίζονται αποτελέσματα.

Αποτελέσματα εκτέλεσης του ερωτήματος E:

```
 Erotima E
Enter Employee AFM:
55241214
```

First Name	Last Name	Car Warehouse ID	Start Date	End Date
Stuart	Clack	428	2018-05-29	null
Stuart	Clack	496	2018-05-26	null
Stuart	Clack	424	2018-05-27	null
Stuart	Clack	355	2018-05-28	null

Αποτελέσματα εκτέλεσης του ερωτήματος F:

Erotima F				
Title	Plate	First Name	Last Name	Service Count
NSX	(CRN,5188)	Dunstan	Josovitz	6
- ES 250	(SCI,8118)	Shelli	Briereton	5
Eagle	(XIM,2533)	Kent	Iohananof	5
Quattro	(DCY,6257)	Meredith	Behan	4
4000	(MJQ,9342)	Winfred	Trevers	4
allroad	(YYE,6740)	Cookie	Saleway	4
CL Models (4)	(YSR,4400)	Rodina	Ghidoni	3
RDX	(TRX,2756)	Mario	Fursey	3
RDX	(XWV,9116)	Shannen	Karlowicz	3
RL Models (2)	(MRQ,3763)	Damon	Hagyard	3
RS 4	(DXE,1904)	Tarah	Boteman	3
Vanquish	(XZR,1880)	Cristine	Sibson	3
- 2.2CL	(OON,4321)	Camala	Oswal	3
NSX	(TYV,5149)	Hollie	Test	3
Legend	(WGW,2883)	Cristine	Sibson	3
Legend	(GLV,4860)	Leia	Klemke	3