

21/6/2020



NTUA-Market

ΑΚΑΔΗΜΑΪΚΟ
ΕΤΟΣ: 2019-
2020

ΒΑΣΕΙΣ ΔΕΔΟΜΕΝΩΝ

Εξαμηνιαία εργασία - Ομάδα 3

| Ζάρα Στέλλα(03117154)

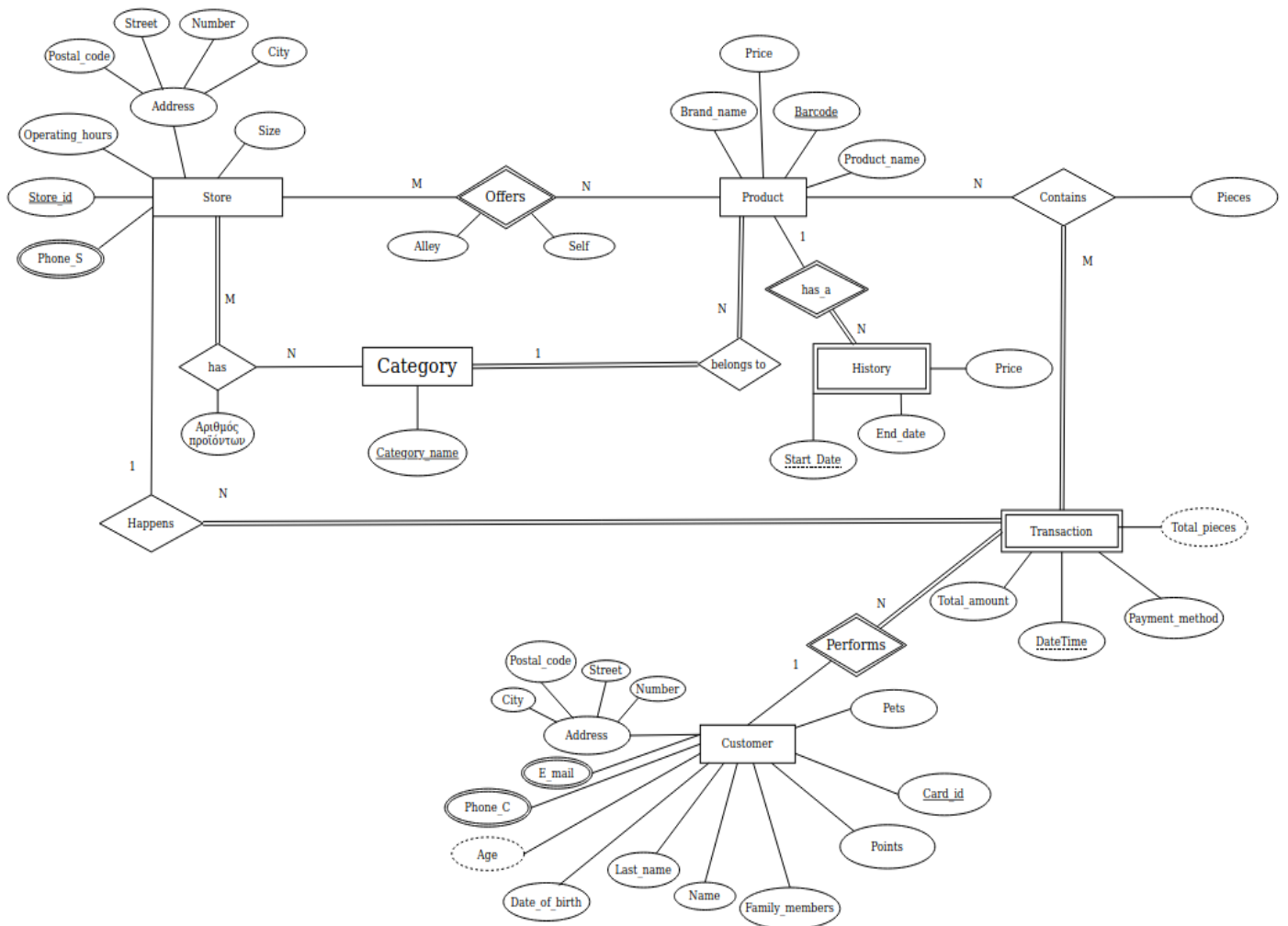
Καπαρού Αλεξάνδρα (03117100)

Λιάγκα Κατερίνα(03117208)

Το βίντεο θα ανέβει από την Αλεξάνδρα Καπαρού(03117100)
στο Microsoft Teams

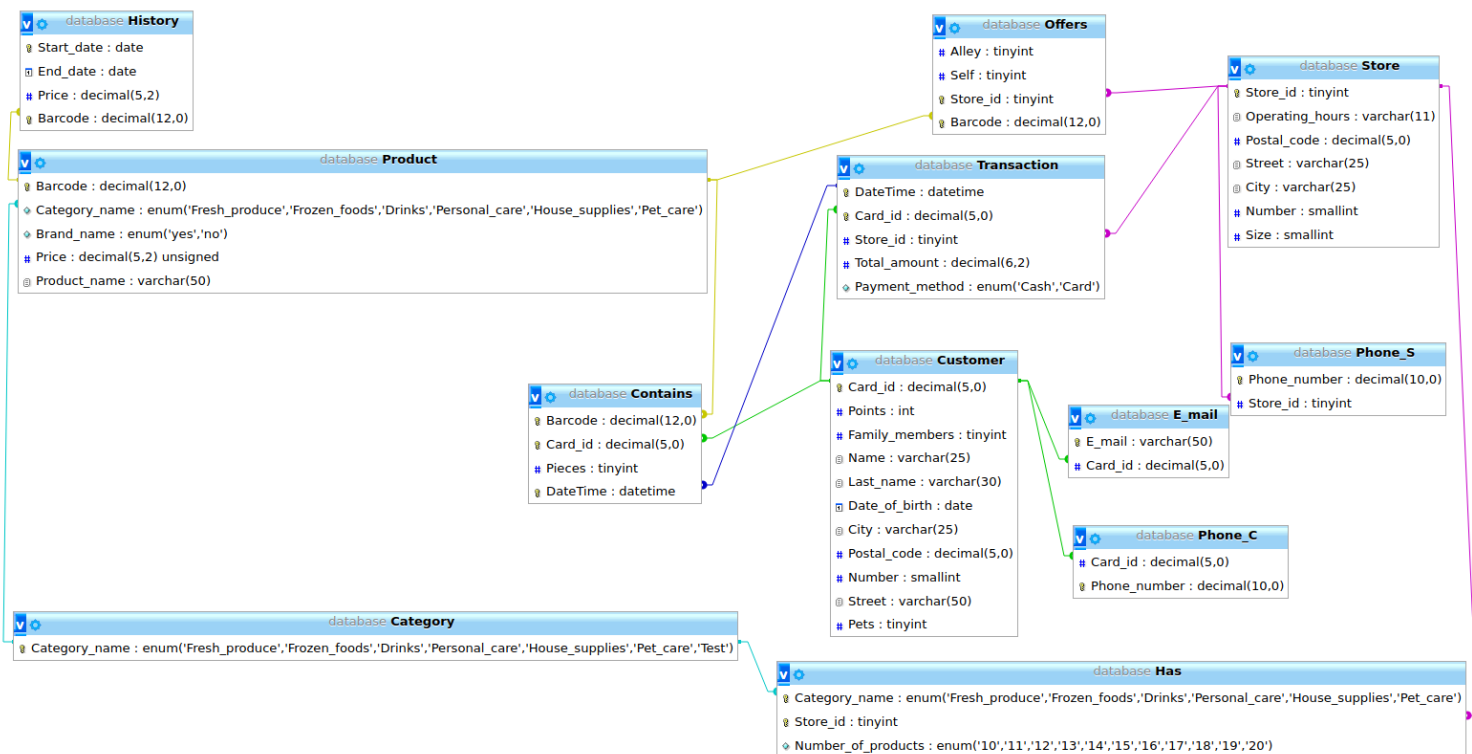
1.ΣΧΕΣΙΑΚΟ ΔΙΑΓΡΑΜΜΑ/ER DIAGRAM

Το ER diagram βάσει του οποίου υλοποιήσαμε την βάση δεδομένων μας έχει ως εξής:



το οποίο έχει διαφορές σε σχέση με αυτό που παραδόθηκε στην πρώτη φάση παράδοσης καθώς θεωρήσαμε πως το αρχικό δεν θα ήταν λειτουργικό στην υλοποίηση.

Βάσει αυτού δημιουργήσαμε το παρακάτω ανάλογο σχεσιακό διάγραμμα απευθείας από το phpmyadmin:



1. Αρχικά θεωρήθηκε, μιας και μιλάμε για αλυσίδα καταστημάτων, ότι το ιστορικό των τιμών θα είναι κοινό για όλα τα επιμέρους καταστήματα, δηλαδή η τιμή ενός προϊόντος θα καθορίζεται κεντρικά και όχι ξεχωριστά από το κάθε κατάστημα.
2. Για να υπάρχει ένα ιστορικό θα πρέπει να συνδέεται με ένα προϊόν.
3. Ένα κατάστημα θα πρέπει να συνδέεται με τουλάχιστον μία κατηγορία.

4. Μία κατηγορία θα πρέπει να περιέχει τουλάχιστον ένα προϊόν και κάθε προϊόν θα πρέπει να ανήκει σε μία μόνο κατηγορία.
5. Για να υπάρχει καταγραφή μιας συναλλαγής στην βάση θα πρέπει αυτή να συνδέεται οπωσδήποτε με ένα τουλάχιστον προϊόν, μόνο ένα κατάστημα, στο οποίο διεξήχθη, και τον πελάτη που την έκανε.
6. Η ηλικία του πελάτη καθώς και ο συνολικός αριθμός των προϊόντων μιας συναλλαγής ορίστηκαν ως derived attributes, το πρώτο θα υπολογίζεται από την ημερομηνία γέννησης του πελάτη και το δεύτερο από το άθροισμα των επιμέρους τεμαχίων κάθε προϊόντος (attribute Pieces στον πίνακα Contains) που συνδέονται με την συγκεκριμένη συναλλαγή.
7. Τα τηλέφωνα τόσο του καταστήματος όσο και του πελάτη, καθώς και τα email του κάθε πελάτη ορίστηκαν ως multivalued attributes, θεωρώντας ότι θα μπορούν να έχουν από καμία έως και πολλαπλές τιμές.

- Περιορισμοί ακεραιότητας:

Αρχικά η mysql κατά την δημιουργία κάθε γνωρίσματος **κάθε πίνακα** ορίζει by default **not null τιμές** κάτι το οποίο δεν αλλάξαμε **εκτός** από το **Product_name** του πίνακα Product, στην οποία η default τιμή είναι το NULL αλλά και από την ημερομηνία στον πίνακα history, στο attribute **End_date** μιας και θέλαμε κατά την εισαγωγή ενός νέου προϊόντος να δημιουργείται αυτόματα ένα ιστορικό για το προϊόν αυτό χωρίς όμως να υπάρχει προκαθορισμένη End_date. Επίσης, επειδή τα στοιχεία κάποιων πινάκων δεν μπορούν να προσδιοριστούν μοναδικά από ένα μόνο attribute, έχουν οριστεί, όπως προβλέπεται και από το ERD, **complex primary keys**. Κάθε attribute που συμπεριλαμβάνεται σε complex primary key αναφέρεται παρακάτω ως partial key. Τέλος, πρέπει να αναφερθεί ότι κάθε primary key συμπεριλαμβανομένων και των complex primary keys είναι **unique**, δηλαδή δεν μπορούμε να έχουμε δύο δεδομένα tuples με

ίδιο primary key. Συγκεκριμένα για τα complex primary keys μοναδικός είναι ο συνδυασμός (η n-άδα) όλων των attributes που είναι partial keys και όχι το κάθε partial key ξεχωριστά.

Όσον αφορά τα **foreign keys** έχουν οριστεί με **on delete cascade** και **on update cascade**, έτσι ώστε να επιτρέπεται διαγραφή και ανανέωση στους πίνακες "γονείς". Μάλιστα όταν γίνεται κάποια διαγραφή ή ανανέωση τιμής σε κάποιο δεδομένο πίνακα γονέα, να διαγράφονται-ανανεώνονται αυτόματα και τα αντίστοιχα δεδομένα που συνδέονταν μέσω foreign keys με τα primary keys των δεδομένων που διαγράφηκαν-ανανεώθηκαν. Συγκεκριμένα για το delete θεωρήθηκε ότι δεν έχει νόημα να κρατούνται δεδομένα για πλέον μη υπάρχοντα καταστήματα ή πελάτες που ακυρώνουν τις κάρτες τους, άρα δεν επισκέπτονται πια την αλυσίδα.

Τα primary και foreign keys που έχουμε ανά πίνακα παρουσιάζονται αναλυτικά παρακάτω:

→ *Customer:*

Card_id : **primary key** επειδή προσδιορίζει μοναδικά τον κάθε πελάτη

→ *E_mail:*

E_mail: **primary key** μιας και στο ERD το E_mail είναι multivalued attribute και είναι μοναδικό για κάθε πελάτη, αφού η πλατφόρμα των email δεν σου επιτρέπει να έχεις την ίδια ακριβώς διεύθυνση με άλλον χρήστη
Card_id: **foreign key** στον πίνακα Customer

→ *Phone_C:*

Card_id: **foreign key** στον πίνακα Customer
Phone_number: **primary key** μιας και στο ERD το Phone_C είναι multivalued attribute και κάθε αριθμός τηλεφώνου είναι μοναδικός για κάθε χρήστη

→ *Transaction*

Το transaction είναι weak entity στο ERD επομένως κάθε του entry πρέπει να προσδιορίζεται από την σχέση που έχει με

άλλους πίνακες στην βάση. Στην προκειμένη περίπτωση αυτοί οι άλλοι πίνακες είναι ο Customer, το Product και το Store

DateTime: **partial key** μιας και το Transaction είναι weak entity

Card_id: **foreign key** στον πίνακα Customer

Store_id: **foreign key** στον πίνακα Store

→ *Product:*

Barcode: **primary key** επειδή προσδιορίζει μοναδικά το κάθε προϊόν

Category_name: **foreign key** στο Category

→ *Contains:*

Barcode: **partial foreign key** στο Product

Card_id: **partial foreign key** στο Customer

DateTime: **partial foreign key** στο Transaction

→ *History:*

Start_date: **partial key** μιας και το History είναι weak entity

Barcode: **foreign key** στο Product

→ *Category:*

Category_name: **primary key** επειδή προσδιορίζει μοναδικά την κάθε κατηγορία

→ *Has:*

Category_name: **foreign key** στο Category

Store_id: **foreign key** στο Store

→ *Store:*

Store_id: **primary key** επειδή προσδιορίζει μοναδικά το κάθε κατάστημα

→ *Phone_S:*

Store_id: **foreign key** στο Store

Phone_number: **partial key** μιας και στο ERD το Phone_S είναι multivalued attribute και κάθε αριθμός τηλεφώνου είναι μοναδικός για κάθε χρήστη

→ *Offers:*

Store_id: **foreign key** στο Store

Barcode: **foreign key** στο Product

- Default τιμές και τύποι attributes για κάθε πίνακα:

Αξίζει να αναφερθεί εκ των προτέρων ότι θεωρούμε πως οι τιμές των προϊόντων έχουν μέχρι 3 ψηφία καθώς και δύο δεκαδικά, ενώ το συνολικό ποσό μιας συναλλαγής μπορεί να έχει μέχρι 4 κανονικά και δύο δεκαδικά ψηφία. Κάθε barcode θα έχει ακριβώς 12 ψηφία, ενώ το Card_id 5 ψηφία και το Store_id θα είναι σειριακά αυξανόμενος αριθμός ξεκινώντας από το 1. Ακόμα, στις περιπτώσεις που θέλαμε να έχουμε επιλογές από ένα συγκεκριμένο σύνολο τιμών χρησιμοποιήσαμε τον τύπο enum. Περισσότερες πληροφορίες παρουσιάζονται αναλυτικά παρακάτω για κάθε attribute κάθε πίνακα:

1. Customer

Ο πίνακας customer έχει τις εξής στήλες:

- Card_id: decimal(5,0) η τιμή του πεδίου αυτού ελέγχεται από *triggers*
- Points: int DEFAULT '0' η τιμή του πεδίου αυτού ελέγχεται από *triggers*
- Family_members: tinyint DEFAULT '0' η τιμή του πεδίου αυτού ελέγχεται από *triggers*
- Name: varchar(25)
- Last_name: varchar(30)
- Date_of_birth: date
- City: varchar(25)
- Postal_code: decimal(5,0) η τιμή του πεδίου αυτού ελέγχεται από *triggers*
- Number: smallint η τιμή του πεδίου αυτού ελέγχεται από *triggers*
- Street: varchar(50)
- Pets: tinyint DEFAULT '0' η τιμή του πεδίου αυτού ελέγχεται από *triggers*

2. E_mail

Επειδή κάθε customer μπορεί να έχει πολλά emails(multivalued), ορίζουμε τον πίνακα E_mail:

- E_mail: varchar(50), η τιμή του πεδίου αυτού ελέγχεται από *triggers*
- Card_id: decimal(5,0) , η τιμή του πεδίου αυτού ελέγχεται από *triggers*

3. Phone_C

Επειδή κάθε customer μπορεί να έχει πολλά phones(multivalued), ορίζουμε τον πίνακα Phone_C:

- Card_id: decimal(5,0), η τιμή του πεδίου αυτού ελέγχεται από *triggers*
- Phone_number: decimal(10,0), η τιμή του πεδίου αυτού ελέγχεται από *triggers*

4. Transaction

- DateTime: datetime, η τιμή του πεδίου αυτού ελέγχεται από *triggers*
- Card_id: decimal(5,0), η τιμή του πεδίου αυτού ελέγχεται από *triggers*
- Store_id: tinyint
- Total_amount: decimal(6,2), η τιμή του πεδίου αυτού ελέγχεται από *triggers*
- Payment_method: enum('Cash','Card') DEFAULT 'Cash'

5. Product

- Barcode: decimal(12,0), η τιμή του πεδίου αυτού ελέγχεται από *triggers*
- Category_name: enum('Fresh_produce', 'Frozen_foods', 'Drinks', 'Personal_care', 'House_supplies', 'Pet_care')
- Price: decimal(5,2) UNSIGNED

- Brand_name: enum('yes','no') DEFAULT 'no'
- Product_Name: varchar(50)

6. Contains

- Barcode: decimal(12,0)
- Card_id: decimal(5,0), η τιμή του πεδίου αυτού ελέγχεται από *triggers*
- Pieces: tinyint DEFAULT '1'
- DateTime: datetime

7. History

Έχουμε φτιάξει ένα complex partial key(μιας και ο πίνακας History είναι weak entity)

- Start_date: date
- End_date: date
- Price: decimal(5,2)
- Barcode: decimal(12,0)

8. Category

- Category_name: enum('Fresh_produce', 'Frozen_foods', 'Drinks', 'Personal_care', 'House_supplies', 'Pet_care')

9. Has

- Category_name: enum('Fresh_produce', 'Frozen_foods', 'Drinks', 'Personal_care', 'House_supplies', 'Pet_care')
- Store_id: tinyint
- Number_of_products: enum('10', '11', '12', '13', '14', '15', '16', '17', '18', '19', '20') DEFAULT '10'

10. Store

- Store_id: tinyint, AUTOINCREMENT
- Operating_hours: varchar(11)

- Postal_code: decimal(5,0), η τιμή του πεδίου αυτού ελέγχεται από *triggers*
- Number: smallint, η τιμή του πεδίου αυτού ελέγχεται από *triggers*
- Street: varchar(25)
- City: varchar(25)
- Size: smallint

11. Phone_S

Επειδή κάθε store μπορεί να έχει πολλά phones(multivalued), ορίζουμε τον πίνακα Phone_S:

- Store_id: tinyint
- Phone_number: decimal(10,0), η τιμή του πεδίου αυτού ελέγχεται από *triggers*

12. Offers

- Alley: tinyint
- Self: tinyint
- Store_id: tinyint
- Barcode: decimal(12,0)

- Triggers:

1. check_for_contains_on_insert: το trigger αυτό αποτρέπει την εισαγωγή στον πίνακα contains αν το barcode έχει λιγότερα από 12 ψηφία, αν το card_id έχει λιγότερα από 5 ψηφία ή αν ο αριθμός των μονάδων ανά προϊόν είναι μικρότερος του 1.
2. check_for_contains_on_update: το trigger αυτό αποτρέπει την ανανέωση στοιχείων στον πίνακα contains αν το barcode έχει λιγότερα από 12 ψηφία, αν το card_id έχει λιγότερα από 5 ψηφία ή αν ο αριθμός των μονάδων ανά προϊόν είναι μικρότερος του 1.

3. **check_for_customer_email_on_insert**: το trigger αυτό αποτρέπει την εισαγωγή στον πίνακα E_mail αν το E_mail δεν έχει την μορφή x...x@y...y.z...z
4. **check_for_customer_email_on_update**: το trigger αυτό αποτρέπει την ανανέωση στοιχείων στον πίνακα E_mail αν το E_mail δεν έχει την μορφή x...x@y...y.z...z
5. **check_for_customer_phone_on_insert**: το trigger αυτό αποτρέπει την εισαγωγή στον πίνακα Phone_C αν το Phone_number έχει λιγότερα από 10 ψηφία.
6. **check_for_customer_phone_on_update**: το trigger αυτό αποτρέπει την ανανέωση στοιχείων στον πίνακα Phone_C αν το Phone_number έχει λιγότερα από 10 ψηφία.
7. **check_for_customer_on_insert**: το trigger αυτό αποτρέπει την εισαγωγή στον πίνακα Customer αν ο αριθμός των points είναι μικρότερος του 0, αν το card_id έχει λιγότερα από 5 ψηφία, αν ο αριθμός των Family_members είναι μικρότερος του 0, αν το Postal_code έχει λιγότερα από 5 ψηφία ή αν ο αριθμός του Number ή του Pets είναι μικρότερος του 0.
8. **check_for_customer_on_update**: το trigger αυτό αποτρέπει την ανανέωση στοιχείων στον πίνακα Customer αν ο αριθμός των points είναι μικρότερος του 0, αν το card_id έχει λιγότερα από 5 ψηφία, αν ο αριθμός των Family_members είναι μικρότερος του 0, αν το Postal_code έχει λιγότερα από 5 ψηφία ή αν ο αριθμός του Number ή του Pets είναι μικρότερος του 0.
9. **check_for_history_on_insert**: το trigger αυτό αποτρέπει την εισαγωγή στον πίνακα History αν το price είναι 0 ή αρνητικό, αν το Barcode έχει λιγότερα από 12 ψηφία ή αν η End_date είναι πιο νωρίς από την Start_date.
10. **check_for_history_on_update**: το trigger αυτό αποτρέπει την ανανέωση στοιχείων στον πίνακα History αν το price είναι 0 ή αρνητικό, αν το Barcode έχει λιγότερα από 12 ψηφία ή αν η End_date είναι πιο νωρίς από την Start_date.

11. **check_for_offers_on_insert**: το trigger αυτό αποτρέπει την εισαγωγή στον πίνακα Offers αν το Barcode έχει λιγότερα από 12 ψηφία, αν η τιμή του Alley δεν ανήκει στο διάστημα $[0,10]$ ή αν η τιμή του Self δεν ανήκει στο διάστημα $[0,10]$.
12. **check_for_offers_on_update**: το trigger αυτό αποτρέπει την ανανέωση στοιχείων στον πίνακα Offers αν το Barcode έχει λιγότερα από 12 ψηφία, αν η τιμή του Alley δεν ανήκει στο διάστημα $[0,10]$ ή αν η τιμή του Self δεν ανήκει στο διάστημα $[0,10]$.
13. **check_for_product_on_insert**: το trigger αυτό αποτρέπει την εισαγωγή στον πίνακα Product αν το Barcode έχει λιγότερα από 12 ψηφία ή αν η το price είναι αρνητικό.
14. **check_for_product_on_update**: το trigger αυτό αποτρέπει την ανανέωση στοιχείων στον πίνακα Product αν το Barcode έχει λιγότερα από 12 ψηφία ή αν το price είναι αρνητικό.
15. **check_for_store_on_insert**: το trigger αυτό αποτρέπει την εισαγωγή στον πίνακα Store αν το Postal_code έχει λιγότερα από 5 ψηφία, αν η το size είναι αρνητικό ή 0 ή αν το Number είναι αρνητικό.
16. **check_for_store_on_update**: το trigger αυτό αποτρέπει την ανανέωση στοιχείων στον πίνακα Store αν το Postal_code έχει λιγότερα από 5 ψηφία, αν η το size είναι αρνητικό ή 0 ή αν το Number είναι αρνητικό.
17. **check_for_store_phone_on_insert**: το trigger αυτό αποτρέπει την εισαγωγή στον πίνακα Phone_S αν το Phone_number έχει λιγότερα από 10 ψηφία.
18. **check_for_store_phone_on_update**: το trigger αυτό αποτρέπει την ανανέωση στοιχείων στον πίνακα Phone_S αν το Phone_number έχει λιγότερα από 10 ψηφία.
19. **check_for_transaction_on_insert**: το trigger αυτό αποτρέπει την εισαγωγή στον πίνακα Transaction αν το Card_id έχει λιγότερα από 5 ψηφία ή το Total_amount είναι μικρότερο ή ίσο του 0.

20. **check_for_transaction_on_update**: το trigger αυτό αποτρέπει την εισαγωγή στον πίνακα Transaction αν το Card_id έχει λιγότερα από 5 ψηφία ή το Total_amount είναι μικρότερο ή ίσο του 0.

- Περιορισμοί σε επίπεδο εφαρμογής:

1. Για Update:

Γενικά θα πρέπει να πατήσουμε το τωρινό αναγνωριστικό primary key και στην συνέχεια από κάτω να κάνουμε τις ανανεώσεις που επιθυμούμε.

- Στον πελάτη έχουμε θεωρήσει ότι παραμένουν πάντα σταθερά το ονοματεπώνυμο του (Name, Last_name), το Card_id του και η ημερομηνία γέννησης του.
- Στα καταστήματα παραμένει σταθερό και δεν αλλάζει το Store_id
- Για τα προϊόντα το μόνο attribute που αλλάζει είναι η τιμή, ταυτόχρονα με την οποία ενημερώνεται και το ιστορικό.

2. Όταν κάνω **create Customer** αυτόματα οι πόντοι του πελάτη τίθενται ίσοι με μηδέν.

3. Η τιμή Price του Product, το Total_amount του Transaction, το Alley και το Shelf του Offers, το Number των διευθύνσεων και το Pieces του Contains είναι πάντα **θετικοί αριθμοί**.

4. Για να **απομονώσουμε μία μόνο ημερομηνία (24ωρο)** στα φίλτρα **των συναλλαγών** θα πρέπει να βάλουμε την συγκεκριμένη ημερομηνία ως Start_Date και ως End_Date την ακριβώς επόμενη ημερομηνία.

5. Επίσης στις ημερομηνίες στα φίλτρα θα πρέπει να συμπληρωθούν και το Start_Date και το End_Date προκειμένου να έχουμε αποτελέσματα καθώς επίσης θα πρέπει και το Start_Date να είναι πριν το End_Date.

- Views:

Δημιουργήθηκαν δύο Views, το customers_list με όλα τα στοιχεία των πελατών συμπεριλαμβανομένης και της ηλικίας, των email και των τηλεφώνων τους και το sales το οποίο περιέχει όλες τις αγορές που έχουν γίνει στα καταστήματα, συγκεκριμένα περιλαμβάνει το κατάστημα που έγινε η αγορά, τον πελάτη που την έκανε, την ημερομηνία και ώρα της αγοράς, το εν λόγω προϊόν και τα τεμάχια που αγοράστηκαν από αυτό και τέλος το όνομα της κατηγορίας του προϊόντος. Ξέρουμε ότι τα views είναι ουσιαστικά αποθηκευμένα queries που εκτελούνται κάθε φορά που θέλουμε να τα χρησιμοποιήσουμε, για αυτό και τα views είναι αυτόματα ανανεώσιμα, για αυτό επιλέξαμε να μην τα κάνουμε υπερβολικά πολύπλοκα και να εκτελούμε επιπλέον queries για την εξυπηρέτηση των ερωτημάτων.

b. **INDEXES**

Γνωρίζουμε ότι τα ευρετήρια χρησιμοποιούνται για την ταχύτερη προσπέλαση σε ένα table χωρίς να διαβαστούν όλες οι εγγραφές που περιέχει επιβαρύνουν όμως την βάση όσον αφορά την μνήμη. Γι'αυτόν τον λόγο ορίσαμε ευρετήρια μόνο σε columns που χρησιμοποιούμε συχνά σε queries και που δεν έχουν πολλά insertions. Επιπλέον, η mysql ορίζει από μόνη της όλα τα primary και foreign keys ως ευρετήρια επομένως τα ευρετήρια που υλοποιήσαμε εμείς είναι αρχικά τα τρία παρακάτω:

```
CREATE INDEX Transaction_Pieces ON Contains (Pieces)
```

```
CREATE INDEX Payment_amount ON Transaction (Total_amount)
```

```
CREATE INDEX Cash_or_Card ON Transaction (Payment_method)
```

,τα οποία επιλέχτηκαν για την υλοποίηση των φίλτρων που αφορούν τις αγορές που έχουν γίνει με βάση πολλαπλά κριτήρια. Επίσης έχουμε το:

```
CREATE INDEX Alley_Number ON Offers (Alley)
```

,το οποίο χρησιμοποιούμε αρκετά για το query που αφορά τις δημοφιλείς θέσεις μέσα στο κατάστημα, το:

```
CREATE INDEX Age ON Customer (Date_of_Birth)
```

,το οποίο χρησιμοποιούμε αρκετά για το query που αφορά το ποσοστό των ηλικιακών ομάδων που επισκέπτονται το κατάστημα κάθε ώρα λειτουργίας και τέλος το:

```
CREATE INDEX Brand ON Product (Brand_Name)
```

,επειδή το χρησιμοποιούμε αρκετά για το query που αφορά το ποσοστό ανά κατηγορία προϊόντων που οι χρήστες εμπιστεύονται προϊόντα με ετικέτα του καταστήματος.

c. ΣΥΣΤΗΜΑ ΚΑΙ ΓΛΩΣΣΕΣ ΠΡΟΓΡΑΜΜΑΤΙΣΜΟΥ

Το παρόν project υλοποιήθηκε σε σύστημα Linux Ubuntu 18.04. Για την υλοποίηση της βάσης δεδομένων χρησιμοποιήσαμε MySQL server και phpmyadmin, για την επικοινωνία client-server χρησιμοποιήσαμε Apache Web Server, για το server side χρησιμοποιήσαμε PHP, για το client-side χρησιμοποιήσαμε HTML και για την λειτουργικότητα της εφαρμογής χρησιμοποιήσαμε συμπληρωματικά Javascript και CSS.

d. ΒΗΜΑΤΑ ΓΙΑ ΤΗΝ ΕΓΚΑΤΑΣΤΑΣΗ ΤΗΣ ΕΦΑΡΜΟΓΗΣ

Για να εγκαταστήσει κάποιος την εφαρμογή θα πρέπει να έχει κατεβάσει το Apache 2, το phpMyAdmin 4.9.5, καθώς και την sql version 2019 κατά προτίμηση έχοντας περασμένη την έκδοση linux 18.04 στον υπολογιστή του. Θα πρέπει να περάσει, κάνοντας import, τον κώδικα της βάσης που βρίσκεται στο αρχείο database.sql στο phpMyAdmin, ώστε να δημιουργηθούν οι κατάλληλοι πίνακες και ενδεχομένως, αν επιθυμεί, να περαστούν τα δεδομένα των tuples που έχουμε δημιουργήσει. Τέλος, θα πρέπει να αποθηκεύσει όλα τα αρχεία ιστοσελίδων τύπου html, jsp, php, css καθώς και το logo(που περιέχονται στον φάκελο source code στο zip) στο path var/www/html/database. Για να έχει πρόσβαση στην ιστοσελίδα θα πρέπει αφού ανοίξει την βάση στο phpMyAdmin να ανοίξει έναν browser πχ. το Google Chrome και να πατήσει <http://localhost/database/welcome.jsp>, οπότε θα μεταφερθεί στην αρχική Welcome Page της ιστοσελίδας και από εκεί το πως θα πλοηγηθεί στην ιστοσελίδα περιγράφεται στο ερώτημα 4.

2.ΛΙΣΤΑ ΜΕ ΤΑ DDL

Για εξοικονόμηση χώρου στην αναφορά, η λίστα που περιέχει τα DDL για την κατασκευή της βάσης βρίσκονται στο ίδιο zip αρχείο μέσα στον φάκελο sql code με την ονομασία 'tables-creator.sql'.

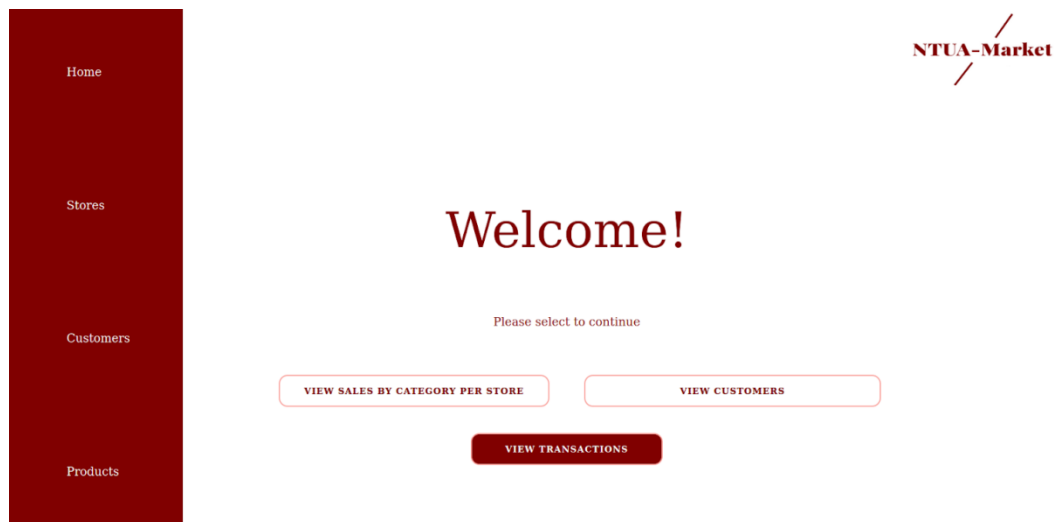
3.SQL ΚΩΔΙΚΑΣ

Όμοια με την λίστα με τα DDL, ο κώδικας για την υποστήριξη της εφαρμογής βρίσκεται στο zip στον φάκελο με την ονομασία 'sql code'.

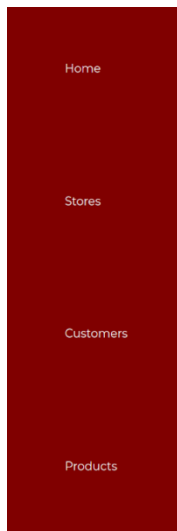
4.ΚΩΔΙΚΑΣ ΓΙΑ ΤΗΝ ΕΓΚΑΤΑΣΤΑΣΗ ΤΗΣ ΕΦΑΡΜΟΓΗΣ

Ο κώδικας για την εγκατάσταση της εφαρμογής βρίσκεται στον φάκελο με το όνομα source code και περιέχει τα κάτωθι αρχεία:

welcome.jsp:



createCustomer.html/createCustomer.php:



Enter a new customer

Card id:
Exactly 5 digits

First name:
Only characters

Last name:
Only characters

Date of birth:
mm/dd/yyyy

City:
Only characters

Postal code:
Exactly 5 digits

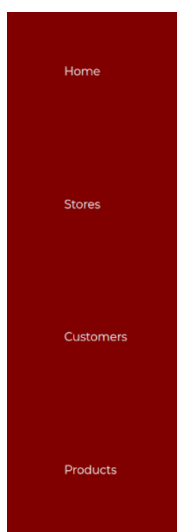
E-mail:
Please give a valid e-mail

Phone number:
Exactly 10 digits

Family members:
Only digits

Number of pets:
Only digits

createProduct.html/createProduct.php:



Enter a new product

Product Name:
Only characters

Barcode:
Exactly 12 digits

Choose Category:
Category Name

Price:
0,00

Is it Brand Name?

createStore.html/createStore.php:

Home

Stores

Customers

Products

NTUA-Market

Enter a new store

Operating Hours:

Insert format HH:MM-HH:MM

Postal Code:

Exactly 5 digits

Street:

Only characters

Street number:

Only digits

updateCustomer.html:

Home

Stores

Customers

Products

NTUA-Market

Update customer

Enter the Card id of the customer you want to update:

Exactly 5 digits

Street:

Only characters

Street number:

Only digits

City:

Only characters

UpdateProduct.html:

Home

Stores

Customers

Products

NTUA-Market

Update an existing product

Enter the Barcode of the product you want to update:

Exactly 12 digits

New Price:

0.00

\$

Submit

Reset

updateStore.html:

Home

Stores

Customers

Products

NTUA-Market

Update store

Enter the Store id of the store you want to update:

Only digits

Operating Hours:

Insert format HH:MM-HH:MM

Postal Code:

Exactly 5 digits

Street:

Only characters

store.php:

Home

Stores

Customers

Products

NTUA-Market

Stores

In order to search for a specific store please select a valid Store id from below

Store id:

Store_id belongs to [1,10]

SubmitReset

Street	Number	City	Operating Hours	Store_id
Rodopis	2	Athens	18:00-21:00	1
G.Averof	36	Thessaloniki	08:00-20:00	2
Kioulara	9	Patra	07:00-21:00	3
Kerkiras	45	Athens	09:00-21:00	4
Eleftherias	11	Patra	08:00-20:00	5
Palaiologou	17	Thessaloniki	08:00-23:00	6
Aristotelous	23	Athens	07:00-23:00	7
Filipou	57	Thessaloniki	8:00-00:00	8
Karatza	29	Patra	9:00-18:00	9
Petras	103	Athens	8:00-22:00	10

Most popular store per city:

Athens

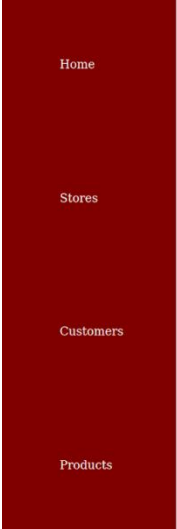
Store_id	Street	Number
1	Rodopis	2

Create Store

customer.php:



products.php



NTUA-Market

Customers

In order to search for a specific customer please select a valid Card id from below

Card id:
Exactly 5 digits

Submit

Reset

Last name	First name	Card id
Alexandrou	Dafni	36749
Antonas	Giannis	19405
Apostolidis	Panayiotis	16203
Apostolotis	Apollon	18954
sa	as	10001
Samiou	Maria	27267
Sarantopoulos	Lefteris	25008
Skorda	Maria	23700
Stratakou	Vasiliki	35278
Tsagari	Danae	21586
Tzini	Theopoula	39875

The customer with the most transactions is:

Card id	Number of transactions
10023	33

Create Customer

Update Customer

NTUA-Market

Products

In order to search for a specific product please select a valid Barcode from below

Barcode:
Exactly 12 digits

Submit

Reset

Category_name	Product_name	Barcode
Fresh_produce	Apples	183740283941
	Berries	729304438277
	Broccoli	157384920424

Home

Stores

Customers

Products

Update Product

Percentage per Product Category that Customers Trust Products with our Store Brand Name:

customer_profile.php:

f_mail(s)
korinaoannidi@outlook.com
korinaofficial@outlook.com

Home

Stores

Customers

Products

NTUA-Market

Customer's profile



The stores this customer has visited:

Store id	Street	Number	City	Postal code
3	Notara	9	Patra	14728
6	Palsiologou	17	Thessaloniki	54236
9	Karatza	29	Patra	26214
10	Petras	103	Athens	10444

The number of stores this customer has visited:

Number of stores this customer visits
4

The 10 most famous products he has bought:

Barcode		
102936400293		
132384738293		
183434546576		
2020	6	7.250000
2020	7	8.875000

The average number of amount in transactions/month:

Year	Month	Average amount
2016	June	73.800000
2020	February	8.875000

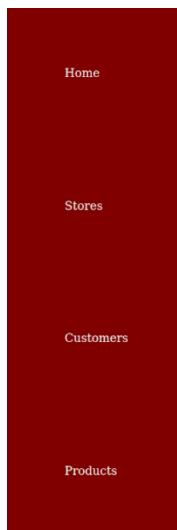
The hour(s) the customer shops from a specific store:

Store 9

Hours in 24-hour format
10
16

Remove Customer

store_profile.php:



Store's profile

Store_id	Operating Hours	Postal Code	Street	City	Number	Size
2	08:00-20:00	11036	G.Averof	Thessaloniki	36	100

Phone number(s)
2103582943

Most popular placements in store:

Barcode	Alley	Shelf
184938205849	2	9
167484635342	2	2
788898399992	2	2

Product ID	Price	Quantity
111113891823	8	3
138408079872	12	5
109878657476	5	1
120499028394	3	3

This store's most profitable hours:

Hour	Total Income
12	45.000000
16	37.150000
11	35.900000
13	28.000000
19	21.400000

Percentage of age range in every operation hour(in 24-hour format):

Hour	Age range	Percentage
10	31-40	100.00%
10	31-40	100.00%
11	41-50	100.00%
12	31-40	100.00%
13	41-50	100.00%
15	21-30	50.00%
	31-40	25.00%
	61-70	25.00%
16	21-30	66.67%
	31-40	33.33%
18	21-30	40.00%
	31-40	40.00%
	41-50	20.00%
19	41-50	100.00%

Remove Store

Update Store

product_profile.php:



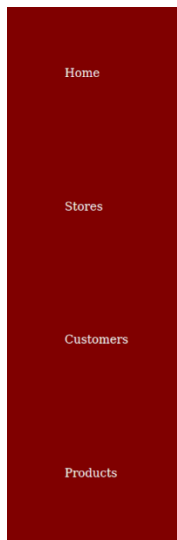
Product's profile

Product info:

Product_name	Barcode	Category	Brand Name	Price(in euros)
Dry Food for Cats	102049737378	Pet_care	yes	5.60

Product History:

Start Date	End Date	Price
2018-04-07	2018-10-07	5.44
2018-10-07	2018-10-21	5.16
2018-10-21	2019-02-13	5.44
2019-02-13	2019-02-27	5.23



Product info:

Product_name	Barcode	Category	Brand Name	Price(In euros)
Dry Food for Cats	102049757378	Pet_care	yes	5.60

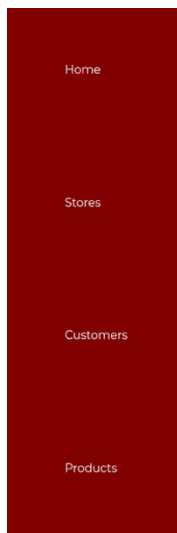
Product History:

Start Date	End Date	Price
2018-04-07	2018-10-07	5.44
2018-10-07	2018-10-21	5.16
2018-10-21	2019-02-13	5.44
2019-02-13	2019-02-27	5.23
2019-02-27	2019-07-26	5.44
2019-07-26		5.60

Remove Product

Update Product

customerWrong.jsp/customerReady.jsp/customerRemove.jsp/customer_updateReady.jsp κτλ:



The customer you entered is not valid

You will be redirected to our input form to try again in 00:05

NavigationViewCustomers.php:



View Customers

Last name	First name	Card ID	Card Points	Age	Date of Birth	City	Street	Number	Postal Code	Family Members	Pets	E-mails	Phone Numbers
Alexandrou	Dafni	36749	987	35	1985-06-20	Patra	Santorinis	69	26425	5	0	dalex@gmail.com	6978654357
Antonas	Giannis	19405	400	46	1974-03-04	Athens	Kastallas	29	11395	0	3	giananton@gmail.gr	6983352758
Apostoliadis	Panayiotis	16203	87	20	1999-09-01	Athens	Andreadou	89	14398	1	0	apostoliadis@hotmail.com	6920586394
Apostolotis	Apollon	18954	964	24	1995-12-09	Patra	Damomos	7	15203	3	0	apollo@gmail.com	6973423536
Athanasopoulou	Emmanouela	19306	16	42	1978-02-02	Patra	Thessalonikis	276	11623	0	3	emma@gmail.com	2140679233
Golaj	Louiza	39852	1198	20	1999-12-03	Athens	Samothrakis	25	11532	2	2	louizag@hotmail.com	6935496461
Hatzidaki	Emmanouela	37854	498	30	1990-03-14	Thessaloniki	Ksenofontos	87	54367	3	2	emmahat@gmail.com	6901230548
Ioannidi	Korina	19320	523	35	1984-11-06	Thessaloniki	Koikimou	8	16325	6	2	korinaioannidi@outlook.com	2103452756
												korinaofficial@outlook.com	
Ioannou	Dimitris	12345	1	26	1994-05-25	Patra	Pantanassis	245	11125	10	1	jimmyioannou@gmail.com	2108394455

transactions.html/php:

Home

Stores

Customers

Products

NTUA-Market

Transactions

Firstly select a Store:

Select a store:

Now you can choose multiple filters to filter the above transactions:

Select a category:

Select a payment method:

Total Amount

Insert total amount in €

Number of pieces

Insert number of pieces

ViewSales.php:

Home

Stores

Customers

Products

NTUA-Market

View Sales per Store

Store ID	Category	Sold Pieces
1	Fresh_produce	16
	Frozen_foods	7
	Drinks	12
	Personal_care	12
	House_supplies	21
	Pet_care	11
2	Fresh_produce	5
	Frozen_foods	6
	Drinks	80
3	Personal_care	31

theme.css: Περιέχει το θέμα που χρησιμοποιήσαμε σε όλες τις σελίδες.

logo.png: Περιέχει το logo της ιστοσελίδας μας.

Καθώς επίσης περιέχονται αρχεία που βοηθούν στην λειτουργικότητα των σελιδών.

~Τέλος αναφοράς~