

# Βάσεις Δεδομένων Project 2022-2023

## Μέλη(με αλφαβητική σειρά επωνύμων):

1) Όνομα: Αγγουρά Ρουμπίνη – Μαρία

AM: 1084634

Έτος Φοίτησης: 3<sup>ο</sup>

2) Όνομα: Παυλόπουλος Ιάσοντας

AM: 1084565

Έτος Φοίτησης: 3<sup>ο</sup>

## Περιεχόμενα:

### ❖ Εισαγωγή

### ❖ Κεφάλαιο 1

- Σχεσιακό Διάγραμμα
- Νέοι Πίνακες
- Τροποποιήσεις σε ήδη υπάρχοντες πίνακες
- Κώδικας για την δημιουργία της βάσης

### ❖ Κεφάλαιο 2

- 3.1.3.1
- 3.1.3.2
- 3.1.3.3
- 3.1.3.4(α)
- 3.1.3.4(β)
- Έξτρα Stored Procedures

### ❖ Κεφάλαιο 3

- 3.1.4.1
- 3.1.4.2
- 3.1.4.3
- Έξτρα Triggers

### ❖ Κεφάλαιο 4

- Εισαγωγή
- Τεκμηρίωση Και Επεξήγηση
- Σενάριο Χρήσης

### ❖ Κεφάλαιο 5

- Κώδικας
- Επεξήγηση Κώδικα

### ❖ Βιβλιογραφία

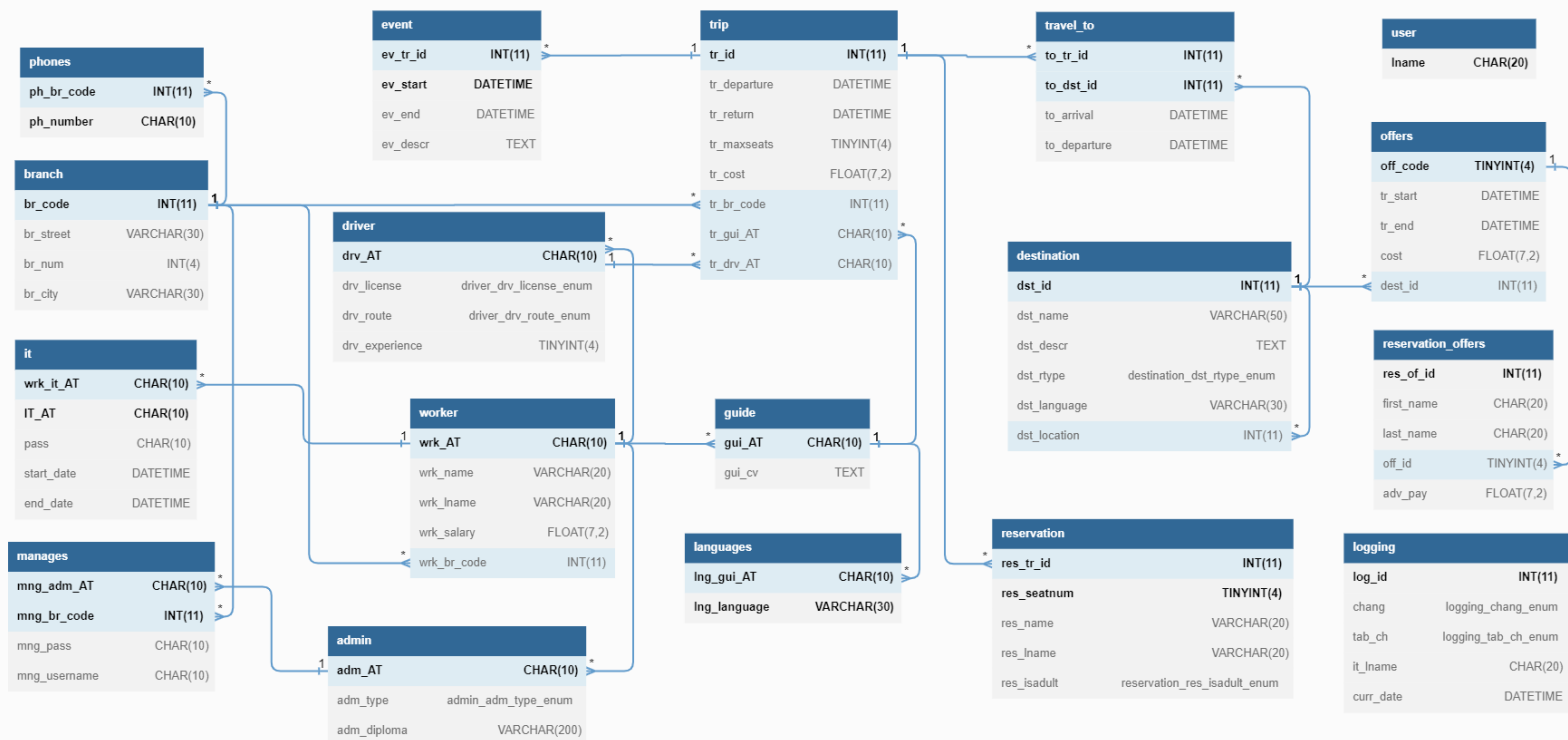
## Εισαγωγή

Πριν ξεκινήσουμε την παρουσίαση του Project, θα θέλαμε να σας ενημερώσουμε πως όλα τα resource files, όπως το αρχείο με τα 60.000 Inserts, τα Icons που χρησιμοποιήσαμε και το αρχείο SQL, συμπεριλαμβανομένης και αυτής της αναφοράς μπορείτε να τα βρείτε στο παρακάτω Github Repository. Επίσης μπορείτε να δείτε τις αλλαγές που έχουν γίνει από την αρχή δημιουργίας του Project (commits).

**Github Repository:** <https://github.com/CallMeJasonYT/DB-Project-2022>

# Κεφάλαιο 1

## Σχεσιακό Διάγραμμα



## Νέοι Πίνακες

- It**: Αντιπροσωπεύει τους workers που ανήκουν στην κατηγορία It Managers. Όπως φαίνεται στο παραπάνω διάγραμμα, ο πίνακας It, αποτελείται από 5 πεδία, εκ των οποίων τα δύο αποτελούν και πρωτεύον κλειδιά του. Τα πεδία του είναι τα εξής:
  - ❖ **wrk\_it\_AT**: Είναι πρωτεύον και ξένο κλειδί που δείχνει στον πίνακα worker και αντιπροσωπεύει τον αριθμό ταυτότητας των εργαζομένων, στην συγκεκριμένη περίπτωση του It Manager.
  - ❖ **IT\_AT**: Είναι πρωτεύον κλειδί του πίνακα μαζί με το wrk\_it\_AT. Αποτελεί το username του IT Manager με το οποίο μπορεί να συνδεθεί στην βάση δεδομένων.
  - ❖ **pass**: Είναι ο κωδικός του It Manager με τον οποίο μπορεί να συνδεθεί στην βάση.
  - ❖ **start\_date**: Αποτελεί την ημερομηνία που ο IT Manager ξεκίνησε να έχει αυτήν την θέση.
  - ❖ **end\_date**: Αποτελεί την ημερομηνία που ο IT Manager σταμάτησε να έχει αυτήν την θέση. Αν ο IT Manager δεν έχει φύγει ακόμα από την εταιρεία, το πεδίο αυτό παίρνει την τιμή null.
- Offers**: Έπειτα, δημιουργήσαμε τον πίνακα offers, στόχος του οποίου είναι να αποθηκεύει offers για καινούρια ταξίδια που δεν έχει δοθεί ακριβής ημερομηνία ακόμα. Ο πίνακας Offers αποτελείται από 5 πεδία:
  - ❖ **off\_code**: Είναι το Πρωτεύον Κλειδί και αντιπροσωπεύει τον αριθμό του offer. Αυξάνεται με auto\_increment.
  - ❖ **tr\_start**: Αποτελεί την ημερομηνία και ώρα έναρξης που μπορεί να πραγματοποιηθεί ένα ταξίδι
  - ❖ **tr\_end**: Αποτελεί την ημερομηνία και ώρα λήξης που μπορεί να πραγματοποιηθεί ένα ταξίδι
  - ❖ **cost**: Αποτελεί το κόστος ανά άτομο για αυτό το offer
  - ❖ **dest\_id**: Αποτελεί τον κωδικό προορισμού. Είναι ξένο κλειδί το οποίο συνδέεται με τον πίνακα destination.
- Reservation\_offers**: Αποτελεί έναν πίνακα στον οποίο ανήκουν όλες οι προσφορές για τα ταξίδια που ανήκουν στον πίνακα offers. Έχει 5 πεδία τα οποία είναι τα εξής:
  - ❖ **res\_of\_id**: Είναι το πρωτεύον κλειδί του πίνακα. Αποτελεί τον μοναδικό κωδικό της προσφοράς, και αυξάνεται με auto\_increment.
  - ❖ **last\_name**: Αποτελεί το επώνυμο του ατόμου που κάνει την προσφορά για κράτηση.
  - ❖ **first\_name**: Αποτελεί το όνομα του ατόμου που κάνει την προσφορά για κράτηση.
  - ❖ **off\_id**: Αποτελεί το id της προσφοράς ταξιδιού στην οποία απευθύνεται η κράτηση. Είναι ξένο κλειδί που συνδέεται με τον πίνακα offers.
  - ❖ **adv\_pay**: Αποτελεί τον προκαταβολή που προσφέρει το άτομο που έκανε την κράτηση.

- Logging: Ο πίνακας logging, είναι απαραίτητος για την αναγνώριση των αλλαγών που γίνονται στην βάση, το είδος τους, και τον χρήστη που τις πραγματοποίησε. Αποτελείται από τα εξής 5 πεδία:
  - ❖ log\_id: Είναι το Πρωτεύον Κλειδί και αντιπροσωπεύει τον αριθμό της αλλαγής ως αναγνωριστικό κωδικό. Αυξάνεται με auto\_increment.
  - ❖ chang: Αποτελεί το είδος της αλλαγής η οποία εκτελείται.
  - ❖ tab\_ch: Αποτελεί το όνομα του πίνακα στον οποίο γίνεται η αλλαγή.
  - ❖ it\_lname: Αποτελεί το επώνυμο του χρήστη ο οποίος κάνει την αλλαγή.
  - ❖ curr\_date: Αποτελεί την ημερομηνία και ώρα κατά την οποία γίνεται η αλλαγή.
- User: Τέλος δημιουργήσαμε τον πίνακα user, ο οποίος χρειάζεται ώστε να αποθηκεύουμε σε κάθε σύνδεση χρήστη, το επώνυμό του. Αποτελείται από το εξής πεδίο:
  - ❖ lname: Είναι το Πρωτεύον Κλειδί και αντιπροσωπεύει το επώνυμο του χρήστη που κάνει login στην βάση.

## Τροποποιήσεις σε ήδη υπάρχοντες πίνακες

- Manages: Στον πίνακα manages προσθέσαμε δύο ακόμα πεδία. Αυτό έγινε για να μπορέσουμε να έχουμε σύνδεση στην βάση και των διευθυντών του κάθε branch.  
Τα καινούρια κλειδιά είναι τα εξής:
  - ❖ mng\_pass: Αποτελεί τον κωδικό του admin για την σύνδεση του στην βάση.
  - ❖ mng\_username: Αποτελεί το username του admin για την σύνδεσή του στην βάση.

## Κώδικας για την δημιουργία της βάσης

Προσοχή!! Στον παρακάτω κομμάτι κώδικα δεν συμπεριλαμβάνουμε τα inserts για τον πίνακα reservation\_offers καθώς είναι 60000 γραμμές κώδικα και δεν γίνεται να είναι σε αυτό το αρχείο. Αυτές οι εγγραφές θα βρίσκονται σε έναν έξτρα αρχείο txt, με το όνομα inserts.txt .

```
DROP database travel_agency;
CREATE DATABASE travel_agency;

USE travel_agency;

CREATE TABLE branch(
  br_code INT(11) NOT NULL AUTO_INCREMENT,
  br_street VARCHAR(30) DEFAULT 'UNKNOWN' NOT NULL,
  br_num INT(4) NOT NULL,
  br_city VARCHAR(30) DEFAULT 'UNKNOWN' NOT NULL,
  PRIMARY KEY(br_code)
);

INSERT INTO branch VALUES
(null, 'Karaiskaki', 13, 'Patra'),
(null, 'Kanakari', 10, 'Patra'),
(null, 'Korinthou', 27, 'Patra'),
(null, 'Zaimi', 30, 'Patra'),
(null, 'Maizonos', 115, 'Patra'),
(null, 'Gounari', 86, 'Patra'),
(null, 'Panourgia', 54, 'Patra'),
(null, 'Aretha', 57, 'Patra'),
(null, 'Ag. Nikolaou', 73, 'Patra'),
(null, 'Kolokotroni', 163, 'Patra');

CREATE TABLE phones(
  ph_br_code INT(11) NOT NULL,
  ph_number CHAR(10) NOT NULL,
  PRIMARY KEY(ph_br_code, ph_number),
  CONSTRAINT phones1 FOREIGN KEY (ph_br_code) REFERENCES branch(br_code)
  ON DELETE CASCADE ON UPDATE CASCADE
);

INSERT INTO phones VALUES
(1, '2104374832'),
(1, '6972345211'),
(2, '2114659823'),
(2, '6984635247'),
(3, '2432233445'),
(3, '6908989786'),
(4, '2103774332'),
(4, '6984756344'),
```

```
(5, '2124621442'),
(5, '6927651853'),
(6, '2152663927'),
(6, '6937462443'),
(7, '2108694523'),
(7, '6932618542'),
(8, '2721074556'),
(8, '6936172534'),
(9, '2710456217'),
(9, '6947527142'),
(10, '2335473923'),
(10, '6936251625');
```

```
CREATE TABLE worker(
  wrk_AT CHAR(10) NOT NULL,
  wrk_name VARCHAR(20) DEFAULT 'UNKNOWN' NOT NULL,
  wrk_lname VARCHAR(20) DEFAULT 'UNKNOWN' NOT NULL,
  wrk_salary FLOAT(7,2) NOT NULL,
  wrk_br_code INT(11) NOT NULL,
  PRIMARY KEY(wrk_AT),
  CONSTRAINT worker1 FOREIGN KEY(wrk_br_code) REFERENCES branch(br_code)
  ON DELETE CASCADE ON UPDATE CASCADE
);
```

```
INSERT INTO worker VALUES
('AM71514316', 'Petros', 'Giorgos', 568.1, 1),
('AW79051091', 'Mario', 'Luigi', 874.7, 2),
('CX68594948', 'Andreas', 'Oikonomou', 1233.2, 3),
('AT53041686', 'Ektoras', 'Samouil', 820.6, 4),
('YS17377807', 'Konstantinos', 'Kostantinou', 889.1, 5),
('CX21653217', 'Iasonas', 'Pavlopoulos', 2776.2, 6),
('XR13453928', 'Andreas', 'Andreadis', 1500.32, 7),
('SA33423559', 'Meletios', 'Polidouris', 1222.44, 8),
('TS73756168', 'Loukas', 'Ramakis', 1000.1, 9),
('YR41149523', 'Axilleas', 'Patroklos', 1022.3, 10),
('RW93324684', 'Roumpini', 'Aggoura', 2620.7, 1),
('WA59455740', 'Maria', 'Louloudi', 433.5, 2),
('SC32758869', 'Prigkipissa', 'Vasilissa', 655.2, 3),
('FG91992776', 'Katerina', 'Papaflessa', 836.44, 4),
('WE17292308', 'Eleni', 'Oikonomakou', 988.2, 5),
('AC18362095', 'Maritini', 'Petroula', 2500.21, 5),
('AY95427438', 'Sevi', 'Papadopolou', 547.2, 6),
('AT13240957', 'Vasiliki', 'Malama', 766.5, 8),
('AT26598124', 'Danae', 'Kathariou', 1923.9, 7),
('WE30613700', 'Anastasia', 'Petropoulou', 957.2, 1),
('AQ88165869', 'Paris', 'Pettas', 1120.1, 9),
('AT23079419', 'Dimitris', 'Petreas', 1200.3, 10),
('TA54439709', 'Efi', 'Persikou', 942.3, 3),
('AY57791065', 'Davon', 'Sampson', 977.53, 4),
('AS33050797', 'Declan', 'Kelly', 1024.82, 5),
('YT13536905', 'Donavan', 'Osborn', 845.9, 6),
('XY88410423', 'Rashad', 'Noble', 1230.23, 7),
('SA88164961', 'Eleni', 'Papadaki', 1422.61, 1),
('AS83636600', 'Dionissis', 'Trivizas', 873.21, 2),
('XH62833740', 'Xristos', 'Petropoulos', 965.22, 3),
('FX23226072', 'Xaralampos', 'Xristidis', 743.1, 9),
('SY94496257', 'Andreas', 'Protopsaltis', 1247.91, 8),
('AW92790994', 'Aggeliki', 'Reka', 998.91, 10),
('SF16806081', 'Zaxarias', 'Michalainas', 1123.9, 7),
('SA47300877', 'Maria', 'Neokosmidi', 1273.85, 10),
('SR77541159', 'Tristin', 'Riley', 1273.85, 2),
('QW82328598', 'Manuel', 'Rowland', 957.21, 4),
('WQ64668093', 'Deanna', 'York', 853.23, 10),
('SS46333586', 'Tommy', 'Carney', 945.12, 6),
('AS85953443', 'Matt', 'Boss', 2931.40, 1),
('AX57737772', 'Gilberto', 'Archer', 2341.23, 1),
('SA98242556', 'Samantha', 'Wilkins', 2422.03, 1),
('SA9824TEST', 'Test', 'User', 2131.03, 1);
```

```
CREATE TABLE it(
  wrk_it_AT CHAR(10) NOT NULL,
  IT_AT CHAR(10) NOT NULL,
  pass CHAR(10) DEFAULT 'password' NOT NULL,
  start_date DATETIME NOT NULL,
  end_date DATETIME,
  PRIMARY KEY(IT_AT, wrk_it_AT),
  CONSTRAINT it1 FOREIGN KEY(wrk_it_AT) REFERENCES worker(wrk_AT)
```

```
ON DELETE CASCADE ON UPDATE CASCADE
);
```

```
INSERT INTO it VALUES
('AS85953443', 'IT01', 'pass0', '2022-01-01 09:00:00', '2022-12-30 09:00:00'),
('AX57737772', 'IT02', 'pass1', '2023-01-01 09:00:00', null),
('SA98242556', 'IT03', 'pass2', '2022-01-02 09:00:00', null);
```

```
CREATE TABLE driver(
    drv_AT CHAR(10) NOT NULL,
    drv_license ENUM('A', 'B', 'C', 'D') NOT NULL,
    drv_route ENUM('LOCAL', 'ABROAD') NOT NULL,
    drv_experience TINYINT(4) NOT NULL,
    PRIMARY KEY(drv_AT),
    CONSTRAINT driver1 FOREIGN KEY(drv_AT) REFERENCES worker(wrk_AT)
    ON DELETE CASCADE ON UPDATE CASCADE
);
```

```
INSERT INTO driver VALUES
('SA88164961', 'C', 'ABROAD', 12),
('WA59455740', 'B', 'LOCAL', 11),
('AS33050797', 'D', 'LOCAL', 28),
('XR13453928', 'D', 'LOCAL', 43),
('SC32758869', 'D', 'LOCAL', 1),
('YT13536905', 'D', 'LOCAL', 2),
('AT53041686', 'B', 'ABROAD', 7),
('SY94496257', 'C', 'LOCAL', 32),
('TS73756168', 'D', 'ABROAD', 15),
('YR41149523', 'C', 'ABROAD', 23),
('SR77541159', 'B', 'ABROAD', 23),
('QW82328598', 'C', 'ABROAD', 23),
('WQ64668093', 'B', 'ABROAD', 23),
('SS46333586', 'B', 'ABROAD', 23);
```

```
CREATE TABLE admin(
    adm_AT CHAR(10) NOT NULL,
    adm_type ENUM('LOGISTICS', 'ADMINISTRATIVE', 'ACCOUNTING') NOT NULL,
    adm_diploma VARCHAR(200) DEFAULT 'UNKNOWN' NOT NULL,
    PRIMARY KEY(adm_AT),
    CONSTRAINT admin1 FOREIGN KEY(adm_AT) REFERENCES worker(wrk_AT)
    ON DELETE CASCADE ON UPDATE CASCADE
);
```

```
INSERT INTO admin VALUES
('WE30613700', 'LOGISTICS', 'Logistics Diploma'),
('YS17377807', 'LOGISTICS', 'Economics Diploma'),
('TA54439709', 'ACCOUNTING', 'Mathematical Engineer Diploma Diploma'),
('SF16806081', 'LOGISTICS', 'Logistics Diploma'),
('SA47300877', 'ACCOUNTING', 'Economics Diploma'),
('RW93324684', 'ADMINISTRATIVE', 'Computer Engineer Diploma'),
('AW79051091', 'ADMINISTRATIVE', 'Mechanical Engineer Diploma'),
('CX68594948', 'ADMINISTRATIVE', 'Chemical Engineer Diploma'),
('AY57791065', 'ADMINISTRATIVE', 'Chemical Engineer Diploma'),
('AC18362095', 'ADMINISTRATIVE', 'Buisness Managment Diploma'),
('AT26598124', 'ADMINISTRATIVE', 'Mathematical Engineer Diploma'),
('SA33423559', 'ADMINISTRATIVE', 'Chemical Engineer Diploma'),
('CX21653217', 'ADMINISTRATIVE', 'Computer Engineer Diploma'),
('AQ88165869', 'ADMINISTRATIVE', 'Buisness Management Diploma'),
('AT23079419', 'ADMINISTRATIVE', 'Computer Engineer Diploma');
```

```
CREATE TABLE manages(
    mng_adm_AT CHAR(10) NOT NULL,
    mng_br_code INT(11) NOT NULL,
    mng_pass CHAR(10) NOT NULL,
    mng_username CHAR(10) NOT NULL,
    PRIMARY KEY(mng_adm_AT, mng_br_code),
    CONSTRAINT manages1 FOREIGN KEY (mng_adm_AT) REFERENCES admin(adm_AT)
    ON DELETE CASCADE ON UPDATE CASCADE,
    CONSTRAINT manages2 FOREIGN KEY (mng_br_code) REFERENCES branch(br_code)
    ON DELETE CASCADE ON UPDATE CASCADE
);
```

```
INSERT INTO manages VALUES
('RW93324684', 1, "mng1", "MNG01"),
('AW79051091', 2, "mng2", "MNG02"),
('CX68594948', 3, "mng3", "MNG03"),
```

```
('AY57791065', 4, "mng4", "MNG04"),
('AC18362095', 5, "mng5", "MNG05"),
('AT26598124', 7, "mng6", "MNG06"),
('SA33423559', 8, "mng7", "MNG07"),
('CX21653217', 6, "mng8", "MNG08"),
('AQ88165869', 9, "mng9", "MNG09"),
('AT23079419', 10, "mng10", "MNG10");
```

```
CREATE TABLE guide(
  gui_AT CHAR(10) NOT NULL,
  gui_cv TEXT,
  PRIMARY KEY(gui_AT),
  CONSTRAINT guide1 FOREIGN KEY (gui_AT) REFERENCES worker(wrk_AT)
  ON DELETE CASCADE ON UPDATE CASCADE
);
```

```
INSERT INTO guide VALUES
('AM71514316', 'Worked as a guide at the Acropolis site for 10 years'),
('FG91992776', 'Worked as a guide at the Santorini island for 5 years'),
('WE17292308', 'Worked as a guide at the Mykonos island for 7 years'),
('AY95427438', 'Worked as a guide at the Acropolis site for 5 years'),
('AT13240957', 'Worked as a guide at the Monemvasia castle for 20 years'),
('XY88410423', 'Worked as a guide at the Acropolis site for 15 years'),
('AS83636600', 'Worked as a guide at the Acropolis site for 1 year'),
('XH62833740', 'Worked as a guide at the Acropolis site for 1 year'),
('FX23226072', 'Worked as a guide at the Acropolis site for 1 year'),
('AW92790994', 'Worked as a guide at the Acropolis site for 1 year');
```

```
CREATE TABLE languages(
  lng_gui_AT CHAR(10) NOT NULL,
  lng_language VARCHAR(30) DEFAULT 'Uknown' NOT NULL,
  PRIMARY KEY(lng_language, lng_gui_AT),
  CONSTRAINT languages1 FOREIGN KEY (lng_gui_AT) REFERENCES guide(gui_AT)
  ON DELETE CASCADE ON UPDATE CASCADE
);
```

```
INSERT INTO languages VALUES
('AM71514316', 'English'),
('AM71514316', 'Greek'),
('FG91992776', 'English'),
('FG91992776', 'Greek'),
('WE17292308', 'English'),
('WE17292308', 'Greek'),
('AY95427438', 'English'),
('AY95427438', 'Greek'),
('AT13240957', 'English'),
('AT13240957', 'Greek'),
('XY88410423', 'English'),
('XY88410423', 'Spanish'),
('XY88410423', 'French'),
('AS83636600', 'English'),
('AS83636600', 'Greek'),
('XH62833740', 'English'),
('XH62833740', 'Greek'),
('XH62833740', 'Italian'),
('XH62833740', 'German'),
('FX23226072', 'English'),
('FX23226072', 'Greek'),
('AW92790994', 'English'),
('AW92790994', 'Greek'),
('AW92790994', 'Spanish'),
('AW92790994', 'French');
```

```
CREATE TABLE trip(
  tr_id INT(11) NOT NULL AUTO_INCREMENT,
  tr_departure DATETIME,
  tr_return DATETIME,
  tr_maxseats TINYINT(4),
  tr_cost FLOAT(7,2),
  tr_br_code INT(11) NOT NULL,
  tr_gui_AT CHAR (10),
  tr_drv_AT CHAR (10),
  PRIMARY KEY(tr_id),
  CONSTRAINT trip1 FOREIGN KEY (tr_br_code) REFERENCES branch(br_code)
  ON DELETE CASCADE ON UPDATE CASCADE,
  CONSTRAINT trip2 FOREIGN KEY (tr_drv_AT) REFERENCES driver(drv_AT)
  ON DELETE CASCADE ON UPDATE CASCADE,
```

```

CONSTRAINT trip3 FOREIGN KEY (tr_gui_AT) REFERENCES guide(gui_AT)
ON DELETE CASCADE ON UPDATE CASCADE
);

```

```

INSERT INTO trip VALUES
(null, '2023-01-29 11:00:00', '2023-02-08 18:00:00', 20, 1023.11, 1, 'AM71514316', 'SA88164961'),
(null, '2023-11-25 10:30:00', '2023-11-30 17:30:00', 5, 361.21, 2, 'AS83636600', 'WA59455740'),
(null, '2023-10-12 11:00:00', '2023-10-16 18:30:00', 15, 451.76, 7, 'XY88410423', 'XR13453928'),
(null, '2023-06-03 12:30:00', '2023-06-15 16:00:00', 6, 842.54, 4, 'FG91992776', 'AT53041686'),
(null, '2023-05-09 11:30:00', '2023-05-20 17:30:00', 9, 822.97, 10, 'AW92790994', 'YR41149523'),
(null, '2023-08-13 11:30:00', '2023-08-23 18:00:00', 50, 677.91, 6, 'AY95427438', 'YT13536905'),
(null, '2023-09-12 12:30:00', '2023-09-17 19:30:00', 30, 532.12, 3, 'XH62833740', 'SC32758869'),
(null, '2023-12-12 10:30:00', '2023-12-20 17:30:00', 15, 742.23, 8, 'AT13240957', 'SY94496257'),
(null, '2023-01-27 12:30:00', '2023-01-30 20:30:00', 17, 352.11, 5, 'WE17292308', 'AS33050797'),
(null, '2023-02-18 11:30:00', '2023-02-28 21:00:00', 20, 874.44, 9, 'FX23226072', 'TS73756168'),
(null, '2023-05-23 11:00:00', '2023-05-30 17:30:00', 10, 858.31, 1, 'AM71514316', 'SA88164961'),
(null, '2023-07-04 12:30:00', '2023-07-17 18:30:00', 4, 923.55, 2, 'AS83636600', 'SR77541159'),
(null, '2023-03-02 12:00:00', '2023-03-08 19:30:00', 22, 690.69, 7, 'XY88410423', 'XR13453928'),
(null, '2023-04-01 11:30:00', '2023-04-06 21:00:00', 14, 570.09, 4, 'FG91992776', 'QW82328598'),
(null, '2023-07-22 12:30:00', '2023-07-28 19:30:00', 5, 700.99, 10, 'AW92790994', 'WQ64668093'),
(null, '2023-09-09 11:30:00', '2023-09-15 18:30:00', 4, 790.89, 6, 'AY95427438', 'SS46333586'),
(null, '2023-09-15 10:30:00', '2023-09-19 19:30:00', 7, 350.99, 3, 'XH62833740', 'SC32758869'),
(null, '2023-12-12 11:30:00', '2023-12-17 17:00:00', 10, 500.34, 8, 'AT13240957', 'SY94496257'),
(null, '2023-10-07 12:30:00', '2023-10-10 17:00:00', 30, 357.10, 5, 'WE17292308', 'AS33050797'),
(null, '2023-12-04 10:30:00', '2023-12-15 18:00:00', 45, 1289.99, 9, 'FX23226072', 'TS73756168'),
(null, '2023-12-04 10:30:00', '2023-12-15 18:00:00', 32, 1289.99, 9, 'FX23226072', 'TS73756168');

```

```

CREATE TABLE event(
  ev_tr_id INT(11) NOT NULL,
  ev_start DATETIME NOT NULL,
  ev_end DATETIME NOT NULL,
  ev_descr TEXT,
  PRIMARY KEY(ev_tr_id, ev_start),
  CONSTRAINT event1 FOREIGN KEY(ev_tr_id) REFERENCES trip(tr_id)
ON DELETE CASCADE ON UPDATE CASCADE
);

```

```

INSERT INTO event VALUES
(1, '2023-01-31 12:00:00', '2023-01-31 14:00:00', 'Visiting The Louvre Museum.'),
(2, '2023-11-27 11:30:00', '2023-11-27 14:30:00', 'Learning about the prosperous history of Kalambaka.'),
(2, '2023-11-28 15:00:00', '2023-11-28 17:30:00', 'Visiting the old part of Kalamata and watching a play at the Castle of Isabeau.'),
(3, '2023-10-13 13:00:00', '2023-10-13 15:30:00', 'Doing a tour of the villages of the island of Corfu!'),
(3, '2023-10-12 19:30:00', '2023-10-12 22:00:00', 'Eating at a local Restaurant.'),
(4, '2023-06-07 13:30:00', '2023-06-07 16:00:00', 'Visiting the Colloseum and the Ancient Market'),
(4, '2023-06-10 15:30:00', '2023-06-10 17:30:00', 'Exploring the canals of the city.'),
(5, '2023-05-13 15:30:00', '2023-05-13 17:30:00', 'Exploring the canals of the city.'),
(6, '2023-08-15 12:30:00', '2023-08-15 13:30:00', 'Visiting the Acropolis and The Acropolis Museum'),
(6, '2023-08-20 12:30:00', '2023-08-20 15:30:00', 'Visiting the White Tower Of Thessaloniki.'),
(7, '2023-09-13 17:30:00', '2023-09-13 19:30:00', 'Visiting the old part of Kalamata and watching a play at the Castle of Isabeau.'),
(8, '2023-12-16 12:30:00', '2023-12-16 15:30:00', 'Visiting the White Tower Of Thessaloniki.'),
(9, '2023-01-28 12:30:00', '2023-01-28 14:30:00', 'Eating at a local Restaurant.'),
(10, '2023-02-23 16:30:00', '2023-02-23 18:00:00', 'Visiting the Barcelona FC Stadium.'),
(10, '2023-02-25 15:00:00', '2023-02-25 17:30:00', 'Visiting a Spectacular small town close to Amsterdam, Delft.'),
(11, '2023-05-24 15:00:00', '2023-05-24 17:30:00', 'Visiting a Spectacular small town close to Amsterdam, Delft.'),
(12, '2023-07-08 16:30:00', '2023-07-08 18:30:00', 'Visiting the Big Ben.'),
(12, '2023-07-11 13:30:00', '2023-07-11 15:30:00', 'Visiting the Berlin Television Tower which is the tallest building in Berlin.'),
(13, '2023-03-03 12:00:00', '2023-03-03 14:30:00', 'Visiting the old part of Kalamata and watching a play at the Castle of Isabeau.'),
(14, '2023-04-02 11:30:00', '2023-04-02 13:00:00', 'Doing a tour of the villages of the island of Corfu!'),
(15, '2023-07-24 15:30:00', '2023-07-24 19:30:00', 'Swimming in the Local Beaches of the Island.'),
(16, '2023-09-11 13:30:00', '2023-09-11 15:30:00', 'Visiting the Berlin Television Tower which is the tallest building in Berlin.'),
(17, '2023-09-16 15:30:00', '2023-09-16 18:30:00', 'Taking a tour around this amazing island.'),
(18, '2023-12-13 16:30:00', '2023-12-13 18:00:00', 'Learning about the prosperous history of Kalambaka.'),
(18, '2023-12-15 12:30:00', '2023-12-15 13:30:00', 'Visiting the Acropolis and The Acropolis Museum.'),
(19, '2023-10-08 12:30:00', '2023-10-08 15:00:00', 'Eating at a local Restaurant.'),
(20, '2023-12-08 12:30:00', '2023-12-15 15:00:00', 'Visiting The Louvre Museum.'),
(20, '2023-12-12 13:30:00', '2023-12-12 16:00:00', 'Visiting the Colloseum and the Ancient Market.'),
(21, '2023-11-04 10:30:00', '2023-11-15 18:00:00', 'Visiting The Louvre Museum.');
```

```

CREATE TABLE destination(
  dst_id INT(11) NOT NULL AUTO_INCREMENT,
  dst_name VARCHAR(50) DEFAULT 'Unknown' NOT NULL,
  dst_descr TEXT,
  dst_rtype ENUM('LOCAL', 'ABROAD'),
  dst_language VARCHAR(30) DEFAULT 'Unknown' NOT NULL,
  dst_location INT(11) NOT NULL,
  PRIMARY KEY(dst_id),
  CONSTRAINT destination1 FOREIGN KEY (dst_location) REFERENCES destination(dst_id)
);

```



```

ON DELETE CASCADE ON UPDATE CASCADE
);

INSERT INTO destination VALUES
(null, 'Kalamata', 'Great place for fans of theatre and the arts to visit. The main draw here is the Castle of Isabeau.', 'LOCAL', 'Greek', 1),
(null, 'Nafpaktos', 'A stunning little port town across the Rio-Antirrio Bridge from Patras. Great place for a weekend trip away from Athens.', 'LOCAL', 'Greek', 2),
(null, 'Corfu', 'It boasts some of the most beautiful beaches in all of Europe.', 'LOCAL', 'Greek', 3),
(null, 'Kalambaka', 'This town is located in Meteora, giving it a stunning landscape as a backdrop to this picturesque city.', 'LOCAL', 'Greek', 4),
(null, 'Heraklion', 'Heraklion, capital of the largest Greek islands, is a great place for those interested in Ancient Greek history.', 'LOCAL', 'Greek', 5),
(null, 'Thessaloniki', 'Thessaloniki, known as the cultural capital of Greece, is also the second-largest city in Greece.', 'LOCAL', 'Greek', 6),
(null, 'Rhodes', 'For those interested in Medieval history. This city is also a UNESCO World Heritage Site.', 'LOCAL', 'Greek', 7),
(null, 'Athens', 'Lots of sightseeing here, especially for those interested in Ancient Greek history', 'LOCAL', 'Greek', 8),
(null, 'Venice', 'Save this one for your one true love, it's a magic spot to share with that special someone.', 'ABROAD', 'Italian', 9),
(null, 'Amsterdam', 'This city is not only one of the most charming cities due to its character and being positioned on the canals, but it is also incredibly fun.', 'ABROAD',
'Dutch', 10),
(null, 'Barcelona', 'It is a hub of new trends in the world of culture, fashion and cuisine.', 'ABROAD', 'Spanish', 11),
(null, 'Rome', 'There's no place like Rome and of course when in Rome, do as the Romans do.', 'ABROAD', 'Italian', 12),
(null, 'Berlin', 'Germany's capital has more history than you can comprehend even after spending weeks in the city.', 'ABROAD', 'German', 13),
(null, 'London', 'There's no other city quite like London and it should go without saying that you simply must visit once in your lifetime.', 'ABROAD', 'English', 14),
(null, 'Paris', 'Paris... a girl's dream. The croissants, crepes, baguettes... yes, this is Paris. You will be in love.', 'ABROAD', 'French', 15),
(null, 'Rome', 'There's no place like Rome and of course when in Rome, do as the Romans do.', 'ABROAD', 'Italian', 16);

CREATE TABLE travel_to(
    to_tr_id INT(11) NOT NULL,
    to_dst_id INT(11) NOT NULL,
    to_arrival DATETIME,
    to_departure DATETIME,
    PRIMARY KEY(to_tr_id, to_dst_id),
    CONSTRAINT travel_to1 FOREIGN KEY (to_tr_id) REFERENCES trip(tr_id)
    ON DELETE CASCADE ON UPDATE CASCADE,
    CONSTRAINT travel_to2 FOREIGN KEY (to_dst_id) REFERENCES destination(dst_id)
    ON DELETE CASCADE ON UPDATE CASCADE
);

INSERT INTO travel_to VALUES
(1, 15, '2023-01-29 15:00:00', '2023-02-08 11:00:00'),
(2, 4, '2023-11-25 14:30:00', '2023-11-27 12:30:00'),
(2, 1, '2023-11-27 17:30:00', '2023-11-30 13:30:00'),
(3, 3, '2023-10-12 17:00:00', '2023-10-14 12:30:00'),
(3, 2, '2023-10-14 18:00:00', '2023-10-16 11:30:00'),
(4, 12, '2023-06-03 16:30:00', '2023-06-09 10:00:00'),
(4, 9, '2023-06-09 16:30:00', '2023-06-15 11:00:00'),
(5, 9, '2023-05-09 15:30:00', '2023-05-20 11:00:00'),
(6, 8, '2023-08-13 15:30:00', '2023-08-19 09:00:00'),
(6, 6, '2023-08-19 15:30:00', '2023-08-23 10:00:00'),
(7, 1, '2023-09-12 16:30:00', '2023-09-17 12:00:00'),
(8, 6, '2023-12-12 14:30:00', '2023-12-20 10:00:00'),
(9, 2, '2023-01-27 16:30:00', '2023-01-30 12:00:00'),
(10, 11, '2023-02-18 15:30:00', '2023-02-24 09:00:00'),
(10, 10, '2023-02-24 14:30:00', '2023-02-28 11:00:00'),
(11, 10, '2023-05-23 16:00:00', '2023-05-30 10:00:00'),
(12, 14, '2023-07-04 16:30:00', '2023-07-10 11:00:00'),
(12, 13, '2023-07-10 14:30:00', '2023-07-17 10:00:00'),
(13, 1, '2023-03-02 15:00:00', '2023-03-08 12:00:00'),
(14, 3, '2023-04-01 14:30:00', '2023-04-06 10:00:00'),
(15, 7, '2023-07-22 15:30:00', '2023-07-28 10:00:00'),
(16, 13, '2023-09-09 13:30:00', '2023-09-15 11:00:00'),
(17, 5, '2023-09-15 14:30:00', '2023-09-19 12:00:00'),
(18, 4, '2023-12-12 16:30:00', '2023-02-14 10:00:00'),
(18, 8, '2023-12-14 12:30:00', '2023-02-17 12:00:00'),
(19, 2, '2023-10-07 15:30:00', '2023-10-10 13:00:00'),
(20, 15, '2023-12-04 13:30:00', '2023-12-11 12:00:00'),
(20, 12, '2023-12-11 14:30:00', '2023-12-15 12:00:00'),
(21, 16, '2023-11-05 10:30:00', '2023-11-05 18:00:00');

CREATE TABLE reservation(
    res_tr_id INT(11) NOT NULL,
    res_seatnum TINYINT(4) NOT NULL,
    res_name VARCHAR(20) DEFAULT 'Uknown' NOT NULL,
    res_lname VARCHAR(20) DEFAULT 'Uknown' NOT NULL,
    res_isadult ENUM('ADULT', 'MINOR'),
    PRIMARY KEY(res_tr_id, res_seatnum),
    CONSTRAINT reservation1 FOREIGN KEY (res_tr_id) REFERENCES trip(tr_id)
    ON DELETE CASCADE ON UPDATE CASCADE
);

INSERT INTO reservation VALUES

```

(1, 15, 'Giorgos', 'Papadopoulos', 'ADULT'),  
 (1, 14, 'Maria', 'Kokkinou', 'ADULT'),  
 (1, 13, 'Takis', 'Papadopoulos', 'MINOR'),  
 (2, 1, 'Giannis', 'Dimitriadis', 'ADULT'),  
 (2, 2, 'Leonidas', 'Dimitriadis', 'MINOR'),  
 (2, 3, 'Tasos', 'Dimitriadis', 'MINOR'),  
 (2, 4, 'Katerina', 'Dimitriadi', 'MINOR'),  
 (2, 5, 'Petroula', 'Asimakopoulou', 'ADULT'),  
 (3, 1, 'Leonidas', 'Asimakopoulos', 'ADULT'),  
 (3, 2, 'Konstantinos', 'Asimakopoulos', 'ADULT'),  
 (4, 5, 'Babis', 'Papageorgiou', 'ADULT'),  
 (4, 6, 'Danae', 'Papageorgiou', 'MINOR'),  
 (5, 8, 'Filia', 'Papakonstantinou', 'ADULT'),  
 (5, 9, 'Maria', 'Papakonstantinou', 'MINOR'),  
 (6, 2, 'John', 'Smith', 'ADULT'),  
 (6, 3, 'Maria', 'Smith', 'MINOR'),  
 (6, 4, 'Natasha', 'Smith', 'MINOR'),  
 (6, 12, 'Clint', 'Smith', 'ADULT'),  
 (6, 13, 'Peter', 'Jones', 'ADULT'),  
 (6, 50, 'Giorgia', 'Dourou', 'ADULT'),  
 (7, 4, 'Roula', 'Andreadi', 'ADULT'),  
 (7, 5, 'Konstantina', 'Andreadi', 'MINOR'),  
 (7, 6, 'Eirini', 'Andreadi', 'MINOR'),  
 (8, 11, 'May', 'Jones', 'ADULT'),  
 (8, 15, 'Sarah', 'Hans', 'ADULT'),  
 (8, 13, 'Eleni', 'Dimoula', 'ADULT'),  
 (8, 14, 'Anastasia', 'Dimoula', 'MINOR'),  
 (9, 15, 'Iasonas', 'Pavlopoulos', 'ADULT'),  
 (9, 14, 'Nefeli', 'Pavlopoulou', 'ADULT'),  
 (9, 8, 'Iasonas', 'Stamoulis', 'ADULT'),  
 (10, 3, 'Roubini', 'Louloudi', 'ADULT'),  
 (10, 4, 'Maria', 'Louloudi', 'MINOR'),  
 (10, 10, 'Princess', 'Violet', 'Adult'),  
 (11, 24, 'Giorgos', 'Papadopoulos', 'ADULT'),  
 (11, 25, 'Petros', 'Papadopoulos', 'MINOR'),  
 (13, 15, 'Giorgos', 'Papandreou', 'ADULT'),  
 (13, 12, 'Tasos', 'Helas', 'ADULT'),  
 (13, 20, 'Anaximandos', 'Asimakopoulos', 'ADULT'),  
 (13, 21, 'Andromaxi', 'Asimakopoulou', 'MINOR'),  
 (14, 3, 'Sam', 'Parker', 'ADULT'),  
 (14, 4, 'Happy', 'Smith', 'ADULT'),  
 (15, 6, 'Maria', 'Papadopoulou', 'ADULT'),  
 (15, 7, 'Nikolas', 'Doukas', 'ADULT'),  
 (15, 1, 'Marios', 'Panagiotou', 'ADULT'),  
 (16, 1, 'Patricia', 'Pilster', 'ADULT'),  
 (16, 2, 'Alice', 'Drake', 'ADULT'),  
 (16, 3, 'Giorgia', 'Dourou', 'MINOR'),  
 (16, 4, 'Jordan', 'Wall', 'MINOR'),  
 (17, 4, 'Jan', 'Greene', 'ADULT'),  
 (17, 5, 'Dylan', 'Mendez', 'MINOR'),  
 (19, 15, 'Kennedy', 'Summers', 'ADULT'),  
 (19, 16, 'Monserrat', 'Carter', 'ADULT'),  
 (19, 17, 'Victoria', 'Holden', 'ADULT'),  
 (20, 3, 'Mitchell', 'Scott', 'ADULT'),  
 (20, 7, 'Trevor', 'Braun', 'ADULT'),  
 (20, 23, 'Jaycee', 'Douglas', 'ADULT'),  
 (20, 14, 'Amir', 'Durham', 'ADULT'),  
 (20, 32, 'Alisha', 'Hunter', 'ADULT'),  
 (20, 28, 'Mateo', 'Garrett', 'ADULT');

```
CREATE TABLE offers(
  off_code TINYINT(4) NOT NULL AUTO_INCREMENT,
  tr_start DATETIME NOT NULL,
  tr_end DATETIME NOT NULL,
  cost FLOAT(7,2) NOT NULL,
  dest_id INT(11) NOT NULL,
  PRIMARY KEY(off_code),
  CONSTRAINT off1 FOREIGN KEY(dest_id) REFERENCES destination(dst_id)
  ON DELETE CASCADE ON UPDATE CASCADE
);
```

```
INSERT INTO offers VALUES
(null, '2023-07-05 09:00:00', '2023-09-27 09:00:00', 456.53, 7),
(null, '2023-11-10 11:00:00', '2023-12-30 11:00:00', 654.53, 10),
(null, '2023-03-08 12:00:00', '2023-05-15 12:00:00', 923.53, 15);
```

```
CREATE TABLE reservation_offers(
```

```
res_of_id INT(11) NOT NULL AUTO_INCREMENT,  
first_name CHAR(20) NOT NULL,  
last_name CHAR(20) NOT NULL,  
off_id TINYINT(4) NOT NULL,  
adv_pay FLOAT(7,2) NOT NULL,  
PRIMARY KEY(res_of_id),  
CONSTRAINT res_off1 FOREIGN KEY(off_id) REFERENCES offers(off_code)  
ON DELETE CASCADE ON UPDATE CASCADE  
);
```

```
CREATE TABLE logging(  
log_id INT(11) NOT NULL AUTO_INCREMENT,  
chang ENUM('INSERT', 'UPDATE', 'DELETE') NOT NULL,  
tab_ch ENUM('trip', 'reservation', 'event', 'travel_to', 'destination') NOT NULL,  
it_lname CHAR(20) NOT NULL,  
curr_date DATETIME NOT NULL,  
PRIMARY KEY(log_id)  
);
```

```
CREATE TABLE user(  
lname CHAR(20) NOT NULL,  
PRIMARY KEY(lname)  
);
```

## Κεφάλαιο 2

### Stored Procedures

#### 3.1.3.1

##### Κώδικας:

```
DROP PROCEDURE IF EXISTS new_driver;
DELIMITER $
CREATE PROCEDURE new_driver(drvAT CHAR(10), first_name VARCHAR(20), last_name VARCHAR(20), salary FLOAT(7,2), license ENUM('A', 'B', 'C', 'D'), dr_route
ENUM('LOCAL', 'ABROAD'), experience TINYINT(4))
BEGIN
    DECLARE not_found INT;
    DECLARE min_br TINYINT;
    DECLARE temp TINYINT;
    DECLARE branches TINYINT;
    DECLARE first_br TINYINT;

    DECLARE drvcursor CURSOR FOR
    SELECT br_code FROM branch;

    DECLARE CONTINUE HANDLER FOR NOT FOUND
    SET not_found=1;
    SET min_br=1;

    SELECT COUNT(*) INTO first_br FROM driver
    INNER JOIN worker ON drv_AT=wrk_AT
    WHERE wrk_br_code=1;

    SET not_found=0;
    OPEN drvcursor;
    REPEAT
        FETCH drvcursor INTO branches;
        IF(not_found=0)
        THEN
            SELECT COUNT(*) INTO temp FROM driver
            INNER JOIN worker ON drv_AT=wrk_AT
            WHERE wrk_br_code=branches;
            IF(temp<first_br)
            THEN
                SELECT branches INTO min_br;
                SET first_br=temp;
            END IF;
        END IF;
    UNTIL(not_found=1)
    END REPEAT;

    INSERT INTO worker VALUES
    (drvAT, first_name, last_name, salary, min_br);
    INSERT INTO driver VALUES
    (drvAT, license, dr_route, experience);
END$
DELIMITER ;
```

**Η παραδοχή μας:** Σε αυτό το stored procedure, ο στόχος είναι η προσθήκη ενός νέου οδηγού(driver) στο υποκατάστημα(branch) που έχει τους λιγότερους.

Παραδοχή που ακολουθήσαμε: Αν υπάρχει παραπάνω από ένα υποκατάστημα με τον μικρότερο αριθμό οδηγών, ο καινούριος οδηγός προστίθεται αυτόματα σε αυτό με το μικρότερο br\_code.

Επίσης θεωρούμε πως από την στιγμή που ζητείται να δοθούν ως ορίσματα, χαρακτηριστικά που ανήκουν στον πίνακα worker, ότι αυτός ο driver δεν έχει εισαχθεί ακόμα ούτε ως worker. Για αυτόν τον λόγο στο τέλος τον εισάγουμε και στον πίνακα worker και στον πίνακα driver.

**Επεξήγηση Κώδικα:** Για την επίτευξη του παραπάνω ζητήματος, δημιουργήσαμε έναν cursor ο οποίος κρατάει τα br\_code από όλα τα branches.

Έπειτα με ένα Select Into Statement, βάλαμε στην τοπική μεταβλητή first\_br, τον αριθμό των drivers που υπάρχουν στο υποκατάστημα με br\_code=1. Αυτό θα το χρησιμοποιήσουμε για να συγκρίνουμε αργότερα το πλήθος των οδηγών των υπόλοιπων υποκαταστημάτων και να κρατήσουμε αυτό με το μικρότερο.

Με την χρήση του cursor λοιπόν περνάμε από όλα τα υποκαταστήματα και μετράμε το πλήθος των οδηγών που ανήκουν σε καθένα από αυτά (count(\*)). Έπειτα, αν το πλήθος αυτό είναι μικρότερο από το “default” που ορίσαμε, δηλαδή αυτό του υποκαταστήματος με br\_code=1, τότε αποθηκεύουμε στην μεταβλητή min\_br το br\_code αυτού του υποκαταστήματος. (Η min\_br έχει αρχικοποιηθεί νωρίτερα με την τιμή 1, ώστε αν δεν βρεθεί branch με μικρότερο αριθμό οδηγών από το “default” να κρατηθεί το 1)

Όταν συγκρίνουμε όλα τα branches, βγαίνουμε από το loop του cursor, και κάνουμε add και στον worker και στον driver τα στοιχεία του driver που δόθηκαν σαν ορίσματα στο stored procedure.

**Παράδειγμα Υλοποίησης:**

Πρώτα ας παρατηρήσουμε το πλήθος των οδηγών σε κάθε υποκατάστημα πριν καλέσουμε την Stored Procedure, με την χρήση ενός Select Statement:

```
SELECT wrk_br_code AS Branch_Code, COUNT(*) AS Number_Of_Drivers FROM driver
INNER JOIN worker ON drv_AT=wrk_AT
GROUP BY wrk_br_code
ORDER BY wrk_br_code;
```

Branch_Code	Number_Of_Drivers
1	1
2	2
3	1
4	2
5	1
6	2
7	1
8	1
9	1
10	2

Έπειτα ας δούμε τους πίνακες worker και driver:  
(Ο πίνακας worker είναι πολύ μεγάλος για να χωρέσει στην αναφορά όποτε θα κάνουμε ένα Select με count για να δούμε πόσοι workers υπάρχουν πριν καλέσουμε την συνάρτηση):

drv_AT	drv_license	drv_route	drv_experience
AS33050797	D	LOCAL	28
AT53041686	B	ABROAD	7
QW82328598	C	ABROAD	23
SA88164961	C	ABROAD	12
SC32758869	D	LOCAL	1
SR77541159	B	ABROAD	23
SS46333586	B	ABROAD	23
SY94496257	C	LOCAL	32
TS73756168	D	ABROAD	15
WA59455740	B	LOCAL	11
WQ64668093	B	ABROAD	23
XR13453928	D	LOCAL	43
YR41149523	C	ABROAD	23
YT13536905	D	LOCAL	2

```
SELECT * FROM driver;
SELECT COUNT(*) FROM worker;
```

COUNT(\*)  
43

Με βάση τα παραπάνω στοιχεία και τις παραδοχές που έχουμε κάνει, παρατηρούμε πως ο καινούριος driver που θα προσθέσουμε θα πρέπει να εισαχθεί στο branch με br\_code = 1.

Καλούμε την Stored Procedure:

```
CALL new_driver('108456523', 'Jason', 'Random', 1234.21, 'C', 'LOCAL', 21);
```

Ας συγκρίνουμε τους πίνακες τώρα με τους αντίστοιχους πίνακες πριν:

Branch_Code	Number_Of_Drivers
1	2
2	2
3	1
4	2
5	1
6	2
7	1
8	1
9	1
10	2

drv_AT	drv_license	drv_route	drv_experience
108456523	C	LOCAL	21
AS33050797	D	LOCAL	28
AT53041686	B	ABROAD	7
QW82328598	C	ABROAD	23
SA88164961	C	ABROAD	12
SC32758869	D	LOCAL	1
SR77541159	B	ABROAD	23
SS46333586	B	ABROAD	23
SY94496257	C	LOCAL	32
TS73756168	D	ABROAD	15
WA59455740	B	LOCAL	11
WQ64668093	B	ABROAD	23
XR13453928	D	LOCAL	43
YR41149523	C	ABROAD	23
YT13536905	D	LOCAL	2

wrk_AT	wrk_name	wrk_lname	wrk_salary	wrk_br_code
108456523	Jason	Random	1234.21	1
AC18362095	Maritini	Petroula	2500.21	5
AM71514316	Petros	Giorgos	568.10	1

COUNT(\*)  
44

Παρατηρούμε λοιπόν απευθείας ότι το branch με br\_code=1, έχει έναν καινούριο driver.  
Από τον πίνακα driver και worker παρατηρούμε πως έχει προστεθεί όντως ο σωστός driver με τα σωστά στοιχεία. Επίσης παρατηρούμε πως όντως αυτός ο worker δεν υπήρχε πριν καλέσουμε την Stored Procedure καθώς βλέπουμε ότι ο αριθμός των workers αυξήθηκε από 43 σε 44.

### 3.1.3.2

#### Κώδικας:

```
DROP PROCEDURE IF EXISTS date_check;
DELIMITER $
CREATE PROCEDURE date_check(br_code INT, date1 DATE, date2 DATE)
BEGIN
    DECLARE dates DATETIME;
    DECLARE tripid INT;
    DECLARE reservations INT;
    DECLARE max_seats tinyINT;
    DECLARE seatdiff tinyINT;
    DECLARE not_found INT;
    DECLARE tridcursor CURSOR FOR
    SELECT tr_id FROM trip WHERE tr_br_code=br_code;
    DECLARE CONTINUE HANDLER FOR NOT FOUND

    SET not_found=1;
    SET not_found=0;

    DROP TABLE IF EXISTS new;
    CREATE TABLE new(
        id INT AUTO_INCREMENT NOT NULL,
        trcost FLOAT(7,2),
        maxseats_ INT,
        reservations_ INT,
        seatdiff_ TINYINT,
        drv_name_ CHAR(20),
        drv_lname_ CHAR(20),
        gui_name_ CHAR(20),
        gui_lname_ CHAR(20),
        tr_dep_ DATETIME,
        tr_ret_ DATETIME,
        PRIMARY KEY(id)
    );

    OPEN tridcursor;
    REPEAT
        FETCH tridcursor INTO tripid;
    IF(not_found=0)
    THEN
        SELECT tr_departure INTO dates FROM trip
        WHERE tr_id=tripid;

        IF (DATEDIFF(dates, date1)>0 && DATEDIFF(dates, date2)<0)
        THEN
            SELECT COUNT(*) INTO reservations FROM reservation
            WHERE tripid=res_tr_id;

            SELECT tr_maxseats INTO max_seats FROM trip
            WHERE tr_id=tripid;

            SET seatdiff=max_seats-reservations;

            INSERT INTO new
            SELECT null, tr_cost, tr_maxseats, reservations,
            seatdiff, a.wrk_name, a.wrk_lname, b.wrk_name,
            b.wrk_lname, tr_departure, tr_return
            FROM trip
            INNER JOIN worker AS a ON tr_drv_AT=a.wrk_AT
            INNER JOIN worker AS b ON tr_gui_AT=b.wrk_AT
            WHERE tr_id=tripid;
        END IF;
    END IF;
    UNTIL(not_found=1)
    END REPEAT;
    SELECT * FROM new;
END$
DELIMITER ;
```

**Επεξήγηση Κώδικα:** Αρχικά δημιουργούμε έναν cursor, ο οποίος κρατάει τα tr\_ids των trips τα οποία διοργανώνονται από το υποκατάστημα που έχει δοθεί ως όρισμα. Δημιουργούμε έναν πίνακα στον οποίο θα αποθηκεύσουμε όλες τις τιμές που θα πάρουμε μέσα από το repeat του cursor ώστε να μπορέσουμε να τις τυπώσουμε όλες στο τέλος σε έναν πίνακα. Έπειτα με repeat του cursor, περνάμε για καθένα από αυτά τα ταξίδια την τιμή tr\_departure τους, στην μεταβλητή dates. Στην συνέχεια με την χρήση ενός If, ελέγχουμε αν αυτή η ημερομηνία βρίσκεται ανάμεσα στις δύο που δόθηκαν στην αρχή. Αν βρίσκεται, μετράμε πόσα reservations έχουν γίνει σε αυτό το ταξίδι και αποθηκεύουμε την τιμή αυτή στην μεταβλητή reservations. Επιπρόσθετα, αποθηκεύουμε στην μεταβλητή max\_seats, τον μέγιστο αριθμό θέσεων που υποστηρίζει το ταξίδι και πραγματοποιούμε και μία αφαίρεση για να υπολογίσουμε πόσες από αυτές τις θέσεις είναι ακόμα κενές. Τέλος περνάμε όλες τις πληροφορίες στον πίνακα new. Βγαίνουμε από το repeat του cursor και κάνουμε έναν SELECT \* στον πίνακα new για να τυπωθούν τα επιθυμητά αποτελέσματα.

**Παράδειγμα Υλοποίησης:**

Αρχικά ας δούμε τον πίνακα trip για να δούμε αναλυτικά όλα τα ταξίδια και τις ημερομηνίες αναχώρησης του καθενός από αυτά:

tr_id	tr_departure	tr_return	tr_maxseats	tr_cost	tr_br_code	tr_gui_AT	tr_drv_AT
1	2023-01-29 11:00:00	2023-02-08 18:00:00	20	1023.11	1	AM71514316	SA88164961
2	2023-11-25 10:30:00	2023-11-30 17:30:00	5	361.21	2	AS83636600	WA59455740
3	2023-10-12 11:00:00	2023-10-16 18:30:00	15	451.76	7	XY88410423	XR13453928
4	2023-06-03 12:30:00	2023-06-15 16:00:00	6	842.54	4	FG91992776	AT53041686
5	2023-05-09 11:30:00	2023-05-20 17:30:00	9	822.97	10	AW92790994	YR41149523
6	2023-08-13 11:30:00	2023-08-23 18:00:00	50	677.91	6	AY95427438	YT13536905
7	2023-09-12 12:30:00	2023-09-17 19:30:00	30	532.12	3	XH62833740	SC32758869
8	2023-12-12 10:30:00	2023-12-20 17:30:00	15	742.23	8	AT13240957	SY94496257
9	2023-01-27 12:30:00	2023-01-30 20:30:00	17	352.11	5	WE17292308	AS33050797
10	2023-02-18 11:30:00	2023-02-28 21:00:00	20	874.44	9	FX23226072	TS73756168
11	2023-05-23 11:00:00	2023-05-30 17:30:00	10	858.31	1	AM71514316	SA88164961
12	2023-07-04 12:30:00	2023-07-17 18:30:00	4	923.55	2	AS83636600	SR77541159
13	2023-03-02 12:00:00	2023-03-08 19:30:00	22	690.69	7	XY88410423	XR13453928
14	2023-04-01 11:30:00	2023-04-06 21:00:00	14	570.09	4	FG91992776	QW82328598
15	2023-07-22 12:30:00	2023-07-28 19:30:00	5	700.99	10	AW92790994	WQ64668093
16	2023-09-09 11:30:00	2023-09-15 18:30:00	4	790.89	6	AY95427438	SS46333586
17	2023-09-15 10:30:00	2023-09-19 19:30:00	7	350.99	3	XH62833740	SC32758869
18	2023-12-12 11:30:00	2023-12-17 17:00:00	10	500.34	8	AT13240957	SY94496257
19	2023-10-07 12:30:00	2023-10-10 17:00:00	30	357.10	5	WE17292308	AS33050797
20	2023-12-04 10:30:00	2023-12-15 18:00:00	45	1289.99	9	FX23226072	TS73756168
21	2023-12-04 10:30:00	2023-12-15 18:00:00	32	1289.99	9	FX23226072	TS73756168

Καλούμε την συνάρτηση:

```
CALL date_check(1, '2023-01-20', '2023-05-20');
```

id	trcost	maxseats_	reservations_	seatdiff_	drv_name_	drv_lname_	gui_name_	gui_lname_	tr_dep_	tr_ret_
1	1023.11	20	3	17	Eleni	Papadaki	Petros	Giorgos	2023-01-29 11:00:00	2023-02-08 18:00:00

Από τον παραπάνω πίνακα παρατηρούμε πως μόνο ένα ταξίδι καλύπτει τις προϋποθέσεις των ορισμάτων της Stored Procedure, και αυτό είναι και αυτό που εμφανίζει και η Stored Procedure. Καλούμε ξανά την Stored Procedure με άλλα ορίσματα για να σιγουρευτούμε ότι δουλεύει για για περισσότερα από 1 ταξίδια:

```
CALL date_check(1, '2023-01-20', '2023-08-20');
```

id	trcost	maxseats_	reservations_	seatdiff_	drv_name_	drv_lname_	gui_name_	gui_lname_	tr_dep_	tr_ret_
1	1023.11	20	3	17	Eleni	Papadaki	Petros	Giorgos	2023-01-29 11:00:00	2023-02-08 18:00:00
2	858.31	10	2	8	Eleni	Papadaki	Petros	Giorgos	2023-05-23 11:00:00	2023-05-30 17:30:00

Παρατηρούμε πως ακόμα και για αυτήν την περίπτωση τα αποτελέσματα είναι τα σωστά.

3.1.3.3

Κώδικας:

```
DROP PROCEDURE IF EXISTS admin_check;
DELIMITER $
CREATE PROCEDURE admin_check(work_name CHAR(20), work_lname CHAR(20))
BEGIN
    DECLARE detail ENUM('LOGISTICS', 'ADMINISTRATIVE', 'ACCOUNTING');
    DECLARE admid CHAR(10);

    SELECT adm_type, adm_AT INTO detail, admid FROM admin
    INNER JOIN worker ON adm_AT=wrk_AT
    WHERE work_name=wrk_name AND work_lname=wrk_lname;

    IF (detail!='ADMINISTRATIVE' AND detail IS NOT NULL)
    THEN
        DELETE FROM admin WHERE adm_AT=admid;
        DELETE FROM worker WHERE wrk_AT=admid;
    ELSEIF (detail IS NULL)
    THEN
        SELECT('This Worker is not an Admin!');
    ELSE
        SELECT('You cannot Delete this User. This User is an Administrative.');
```

**Επεξήγηση Κώδικα:** Αρχικά δημιουργούμε δύο μεταβλητές, την detail και την admid και περνάμε σε αυτές τις τιμές adm\_type και adm\_AT αντίστοιχα (του πίνακα admin) για τις περιπτώσεις που το όνομα και το επώνυμο στον πίνακα admin είναι ίδια με αυτά που δόθηκαν σαν όρισμα στην Stored Procedure.

Έπειτα δημιουργούμε ένα If, και ελέγχουμε αν το type του worker που δόθηκε δεν είναι null και ταυτόχρονα δεν είναι και ‘ADMINISTRATIVE’. Αυτό το κάνουμε για να ελέγξουμε αν ο εργάτης που δόθηκε είναι admin αλλά και ταυτόχρονα αν ανήκει στις κατηγορίες των Admin που μπορούμε να διαγράψουμε ή όχι. Αν λοιπόν δεν είναι null και δεν ανήκει στην κατηγορία ‘ADMINISTRATIVE’ τότε κάνουμε αυτόν τον εργάτης delete και από τον πίνακα worker αλλά και από τον πίνακα admin. Επιπρόσθετα, έχουμε και ένα IF ELSE, το οποίο ελέγχει αν ο εργάτης που δόθηκε σαν όρισμα είναι admin, αν δεν είναι τυπώνει κατάλληλο μήνυμα. Τέλος έχουμε την περίπτωση που ο worker είναι admin και ταυτόχρονα ανήκει στην κατηγορία ‘ADMINISTRATIVE’, τότε τυπώνουμε μήνυμα που λέει ότι δεν μπορεί να διαγραφεί αυτός ο Worker καθώς ανήκει στην παραπάνω κατηγορία.

Παράδειγμα Υλοποίησης:

Πρώτα ας κάνουμε Print όλους τους Admin, πριν καλέσουμε την stored procedure:

adm_AT	adm_type	adm_diploma	wrk_name	wrk_lname
AC18362095	ADMINISTRATIVE	Buisness Managment Diploma	Maritini	Petroula
AQ88165869	ADMINISTRATIVE	Buisness Management Diploma	Paris	Pettas
AT23079419	ADMINISTRATIVE	Computer Engineer Diploma	Dimitris	Petreas
AT26598124	ADMINISTRATIVE	Mathematical Engineer Diploma	Danae	Kathariou
AW79051091	ADMINISTRATIVE	Mechanical Engineer Diploma	Mario	Luigi
AY57791065	ADMINISTRATIVE	Chemical Engineer Diploma	Davon	Sampson
CX21653217	ADMINISTRATIVE	Computer Engineer Diploma	Iasonas	Pavliopoulos
CX68594948	ADMINISTRATIVE	Chemical Engineer Diploma	Andreas	Oikonomou
RW93324684	ADMINISTRATIVE	Computer Engineer Diploma	Roumpini	Aggoura
SA33423559	ADMINISTRATIVE	Chemical Engineer Diploma	Meletios	Polidouris
SA47300877	ACCOUNTING	Economics Diploma	Maria	Neokosmidi
SF16806081	LOGISTICS	Logistics Diploma	Zaxarias	Michalainas
TA54439709	ACCOUNTING	Mathematical Engineer Diplom...	Efi	Persikou
WE30613700	LOGISTICS	Logistics Diploma	Anastasia	Petropoulou
YS17377807	LOGISTICS	Economics Diploma	Konstanti...	Kostantinou

Έπειτα θα κάνουμε 3 διαφορετικές δοκιμές. Η πρώτη θα καλεί την Stored Procedure με ορίσματα που ανήκουν σε Admin τύπου ADMINISTRATIVE. Η δεύτερη θα καλεί την Stored Procedure με ορίσματα Admin που δεν είναι τύπου ADMINISTRATIVE. Και η τρίτη θα καλεί την Stored Procedure με ορίσματα που δεν ανήκουν σε κάποιον Admin.



1° Call:

```
CALL admin_check('Roumpini', 'Aggoura');
```

You cannot Delete this User. This User is an Administrative.

2° Call:

```
CALL admin_check('Anastasia', 'Petrovou');
```

adm_AT	adm_type	adm_diploma	wrk_name	wrk_lname
AC18362095	ADMINISTRATIVE	Buisness Managment Diploma	Maritini	Petroura
AQ88165869	ADMINISTRATIVE	Buisness Management Diploma	Paris	Pettas
AT23079419	ADMINISTRATIVE	Computer Engineer Diploma	Dimitris	Petreas
AT26598124	ADMINISTRATIVE	Mathematical Engineer Diploma	Danae	Kathariou
AW79051091	ADMINISTRATIVE	Mechanical Engineer Diploma	Mario	Luigi
AY57791065	ADMINISTRATIVE	Chemical Engineer Diploma	Davon	Sampson
CX21653217	ADMINISTRATIVE	Computer Engineer Diploma	Iasonas	Pavlopoulos
CX68594948	ADMINISTRATIVE	Chemical Engineer Diploma	Andreas	Oikonomou
RW93324684	ADMINISTRATIVE	Computer Engineer Diploma	Roumpini	Aggoura
SA33423559	ADMINISTRATIVE	Chemical Engineer Diploma	Meletios	Polidouris
SA47300877	ACCOUNTING	Economics Diploma	Maria	Neokosmidi
SF16806081	LOGISTICS	Logistics Diploma	Zaxarias	Michalainas
TA54439709	ACCOUNTING	Mathematical Engineer Diplom...	Efi	Persikou
YS17377807	LOGISTICS	Economics Diploma	Konstanti...	Kostantinou

Παρατηρούμε ότι ο Admin με το όνομα Anastasia Petrovou, δεν υπάρχει πια στην λίστα με τους Admin.

3° Call:

```
CALL admin_check('Petros', 'Giorgos');
```

This Worker is not an Admin!

Παρατηρούμε πως ο εργάτης Petros Giorgos, δεν έχει εμφανιστεί στον πιο πάνω πίνακα, δηλαδή δεν είναι Admin, αρά τυπώνεται σωστά το παραπάνω μήνυμα και δεν επιτρέπει την διαγραφή του.

#### 3.1.3.4 α)

##### Κώδικας:

```
DROP PROCEDURE IF EXISTS prepaid;
DELIMITER $
CREATE PROCEDURE prepaid(cost1 FLOAT(7,2), cost2 FLOAT(7,2))
BEGIN
    DECLARE costs FLOAT(7,2);
    DECLARE res_id INT(11);
    DECLARE not_found INT;

    DECLARE precursor CURSOR FOR
    SELECT res_of_id FROM reservation_offers;
    DECLARE CONTINUE HANDLER FOR NOT FOUND

    SET not_found=1;
    SET not_found=0;

    DROP TABLE IF EXISTS final_paid;
    CREATE TABLE final_paid(
        last_name CHAR(20),
        first_name CHAR(20),
        id INT(11),
        PRIMARY KEY(id)
    );

    OPEN precursor;
    REPEAT
        FETCH precursor INTO res_id;
        IF(not_found=0)
        THEN
            SELECT adv_pay INTO costs FROM reservation_offers
            WHERE res_of_id=res_id;

            INSERT INTO final_paid
            SELECT last_name, first_name, res_of_id FROM reservation_offers
            USE INDEX (reservation_offers_adv_pay_idx)
            WHERE res_of_id=res_id && adv_pay >=cost1 && adv_pay <= cost2;
        END IF;
    UNTIL(not_found=1)
    END REPEAT;
    SELECT last_name AS Last_Name, first_name AS First_Name FROM final_paid;
END$
DELIMITER ;
```

**Επεξήγηση Κώδικα:** Αρχικά δημιουργούμε έναν cursor στον οποίο περνάμε όλα τα res\_od\_id από τον πίνακα reservation\_offers. Έπειτα δημιουργούμε ένα καινούριο table, το final\_raid στο οποίο θα περάσουμε στην συνέχεια όλα τα δεδομένα που θέλουμε να τυπώσουμε, για να τυπωθούν όλα μαζί σε έναν πίνακα. Έπειτα ξεκινάει το repeat του cursor. Σε αυτό περνάμε για κάθε τιμή του cursor, το ποσό που έχουν δώσει ως προκαταβολή, στην μεταβλητή costs. Έπειτα εισάγουμε στον πίνακα final\_raid τα δεδομένα που θα τυπώσουμε. Σε αυτό το κομμάτι χρησιμοποιούμε και τον index, παρόλο που δεν κάνει καμία διαφορά στον χρόνο που χρειάζεται για να εμφανιστούν τα αποτελέσματα. Τέλος βγαίνουμε από το repeat και τυπώνουμε όλο τον πίνακα final\_raid

**Παράδειγμα Υλοποίησης:**

Εδώ καθώς το πλήθος των εγγραφών είναι απίστευτα μεγάλο, δεν γίνεται να έχουμε παράδειγμα πριν την υλοποίηση του stored procedure. Οπότε θα παραθέσουμε μόνο τον πίνακα που τυπώνεται αφού καλέσουμε την συνάρτηση (και αυτόν όχι ολόκληρο)

```
CALL prepaid(100.00,200.00);
```

Last_Name	First_Name
Baldwin	Itzel
Riddle	Jazmine
Baird	Aliyah
Cherry	Joselyn
Baldwin	Kylee
Benton	Augustus
Cox	Gaige
Nolan	Tommy
Townsend	Denzel
Blevins	Milagros
Bryant	Adrien
Morrow	Genesis
Rowland	Manuel
Ward	Layla
Holland	Ishaan
Archer	Gilberto
Sampson	Mckenzie
Cohen	Jovanny
Wilkins	Samantha

3.1.3.4 β)

**Κώδικας:**

```
DROP PROCEDURE IF EXISTS check_offers;
DELIMITER $
CREATE PROCEDURE check_offers(lname CHAR(20))
BEGIN
    DECLARE res_id INT(11);
    DECLARE not_found INT;

    DECLARE ofcursor CURSOR FOR
    SELECT res_of_id FROM reservation_offers WHERE last_name=lname;
    DECLARE CONTINUE HANDLER FOR NOT FOUND

    SET not_found=1;
    SET not_found=0;

    DROP TABLE IF EXISTS new_res_off;
    CREATE TABLE new_res_off(
        offid TINYINT(4),
        resid INT(11),
        first_name CHAR(20),
        last_name CHAR(20),
        PRIMARY KEY(resid)
    );

    OPEN ofcursor;
    REPEAT
        FETCH ofcursor INTO res_id;
        IF(not_found=0)
        THEN
            INSERT INTO new_res_off
            SELECT off_id, res_of_id, first_name, last_name FROM reservation_offers
            WHERE last_name=lname AND res_of_id=res_id;
        END IF;
    UNTIL(not_found=1)
    END REPEAT;
    SELECT offid AS Offer_ID, first_name AS First_Name, last_name AS Last_Name FROM new_res_off
    ORDER BY offid;
END$
DELIMITER ;
```

**Επεξήγηση Κώδικα:** Αρχικά ξεκινάμε δημιουργώντας έναν cursor ο οποίος κρατάει όλα τα res\_of\_it από τον πίνακα reservation\_offers, των οποίων τα lname είναι ίδιο με το lastname που δόθηκε σαν όρισμα στο Stored Procedure. Έπειτα όπως και στο προηγούμενο παράδειγμα δημιουργούμε έναν πίνακα στον οποίο θα αποθηκεύσουμε αργότερα όλες τις τιμές που θα θέλουμε να εκτυπώσουμε, για να μπορέσουμε να τις εμφανίσουμε όλες στον ίδιο πίνακα. Ξεκινάμε το repeat του cursor, και για κάθε διαφορετική τιμή του περνάμε μέσα στον νέο μας πίνακα, τα στοιχεία που ζητούνται από την εκφώνηση. Όταν περάσουμε από όλες τις τιμές του cursor, βγαίνουμε από το repeat και κάνουμε έναν Select για τον πίνακα που δημιουργήσαμε. Επίσης ορίζουμε να εμφανιστούν με βάση το offid τους.

**Παράδειγμα Υλοποίησης:**

Όπως και στο προηγούμενο παράδειγμα, καθώς το πλήθος των εγγραφών είναι απίστευτα μεγάλο, δεν γίνεται να έχουμε παράδειγμα πριν την υλοποίηση του stored procedure. Οπότε θα παραθέσουμε μόνο τον πίνακα που τυπώνεται αφού καλέσουμε την συνάρτηση (και αυτόν όχι ολόκληρο):

```
CALL check_offers('Baldwin');
```

Offer_ID	First_Name	Last_Name
1	Peter	Baldwin
1	Gary	Baldwin
1	Carl	Baldwin
1	Quintin	Baldwin
1	Marissa	Baldwin
1	Ashlee	Baldwin
1	Essence	Baldwin
1	Konner	Baldwin
2	Aidan	Baldwin
2	Audrey	Baldwin
2	Marisa	Baldwin
2	Lance	Baldwin
2	Angelica	Baldwin
2	Kamari	Baldwin
2	Layne	Baldwin
2	Rayne	Baldwin
2	Eduardo	Baldwin
2	Cassandra	Baldwin
2	Alexia	Baldwin

Όπως βλέπουμε τυπώνονται με βάση το offid όπως ακριβώς ζητείται στην εκφώνηση. (Αυτή δεν είναι η αρχή της εκτύπωσης του πίνακα αλλά το κομμάτι στο οποίο βλέπουμε ότι περνάει από τις εγγραφές με offid=1 σε αυτές με offid=2.

**Έξτρα Stored Procedures**

- Κώδικας:**

```
DROP PROCEDURE IF EXISTS seats_trip;
DELIMITER $
CREATE PROCEDURE seats_trip(IN res_id INT, OUT res TINYINT)
BEGIN
    DECLARE currentSeats TINYINT;
    DECLARE rsvd INT;
    DECLARE not_found INT;
    DECLARE max TINYINT;
    DECLARE tripcursor CURSOR FOR
    SELECT res_tr_id FROM reservation WHERE res_tr_id=res_id;
    DECLARE CONTINUE HANDLER FOR NOT FOUND

    SET not_found=1;

    SELECT tr_maxseats INTO max FROM trip WHERE tr_id=res_id;

    SET not_found=0;

    OPEN tripcursor;
    REPEAT
        FETCH tripcursor INTO rsvd;
        IF(not_found=0)
        THEN
            SELECT COUNT(*) INTO currentSeats FROM reservation WHERE res_tr_id=rsvd;

        END IF;
    UNTIL(not_found=1)
    END REPEAT;
    IF (currentSeats < max OR currentSeats IS NULL)
        THEN SET res=1;
        ELSE SET res=0;
    END IF;
END$
DELIMITER ;
```

**Επεξήγηση Κώδικα:** Για την καλύτερη λειτουργία του προγράμματος κρατήσαμε στον κώδικας μας το stored procedure που είχε ζητηθεί να δημιουργήσουμε ως 5<sup>η</sup> εργαστηριακή άσκηση. Αυτό το stored procedure έχει ως στόχο, πριν από κάθε insert στον πίνακα reservation, να ελέγχει αν υπάρχουν διαθέσιμες θέσεις για το ταξίδι στο οποίο αναφέρεται το insert. Έπειτα με ένα trigger που θα αναλύσουμε στην συνέχεια, απαγορεύει το insert σε περίπτωση που δεν υπάρχουν άλλες διαθέσιμες θέσεις.

Για να υλοποιήσουμε αυτό το Stored Procedure, ξεκινάμε δημιουργώντας έναν cursor, που ονομάζουμε tripcursor. Σε αυτόν περνάμε τα res\_tr\_id των reservation που έχουν την ίδια τιμή με το δοσμένο res\_id.

Στην συνέχεια, περνάμε στην μεταβλητή max, το tr\_maxseats από τον πίνακα trip.

Έπειτα ξεκινάμε το repeat του tripcursor και με count(\*) μετράμε όλες τις κρατήσεις που έχουν γίνει για το συγκεκριμένο trip.

Τέλος βγαίνουμε από το repeat και με ένα If Statement, ελέγχουμε αν υπάρχουν διαθέσιμες θέσεις ή όχι. Αν υπάρχουν, θέτουμε την μεταβλητή res=1, ενώ αν δεν υπάρχουν, θέτουμε res=0. Αυτά θα μας χρειαστούν στην συνέχεια στον trigger.

### Παράδειγμα Υλοποίησης:

Καθώς αυτό το Stored Procedure συνδέεται άμεσα με ένα trigger δεν μπορούμε να δούμε την πραγματική του λειτουργία χωρίς αυτό. Παρόλαυτα, μπορούμε να δούμε αν η μεταβλητή res, παίρνει στο τέλος την τιμή που πρέπει να πάρει.

Θα καλέσουμε λοιπόν αυτό το Stored Procedure δύο φορές, μία για ένα ταξίδι που δεν έχει άλλες διαθέσιμες θέσεις, και μία για ένα που έχει.

Για τις ανάγκες του παραδείγματος θα πρέπει να προσθέσουμε στον κώδικα μια έξτρα εντολή select για να εκτυπώσουμε το res στην οθόνη μας.

Ας ξεκινήσουμε με αυτό που δεν έχει άλλες διαθέσιμες θέσεις, το res του θα πρέπει να είναι 0.

```
CALL seats_trip(2,@res);
```

res
0

Και τώρα ας δούμε για ένα trip που έχει ακόμα διαθέσιμες θέσεις.

```
CALL seats_trip(3,@res);
```

res
1

## Triggers

### 3.1.4.1.

#### Κώδικας:

```
DROP TRIGGER IF EXISTS logUTrip;
DELIMITER $
CREATE TRIGGER logUTrip AFTER UPDATE ON trip
FOR EACH ROW
BEGIN
    DECLARE curr_date DATETIME;
    DECLARE last_name CHAR(20);

    SELECT lname INTO last_name FROM user;
    SELECT now() INTO curr_date;
    INSERT INTO logging VALUES
        (null, 'Update', 'trip', last_name, curr_date);
END$
DELIMITER ;

DROP TRIGGER IF EXISTS logDTrip;
DELIMITER $
CREATE TRIGGER logDTrip AFTER DELETE ON trip
FOR EACH ROW
BEGIN
    DECLARE curr_date DATETIME;
    DECLARE last_name CHAR(20);

    SELECT lname INTO last_name FROM user;
    SELECT now() INTO curr_date;
    INSERT INTO logging VALUES
        (null, 'Delete', 'trip', last_name, curr_date);
END$
DELIMITER ;

DROP TRIGGER IF EXISTS logITrip;
DELIMITER $
CREATE TRIGGER logITrip AFTER INSERT ON trip
FOR EACH ROW
BEGIN
    DECLARE curr_date DATETIME;
    DECLARE last_name CHAR(20);

    SELECT lname INTO last_name FROM user;
    SELECT now() INTO curr_date;
    INSERT INTO logging VALUES
        (null, 'Insert', 'trip', last_name, curr_date);
END$
DELIMITER ;

DROP TRIGGER IF EXISTS logURes;
DELIMITER $
CREATE TRIGGER logURes AFTER UPDATE ON reservation
FOR EACH ROW
BEGIN
    DECLARE curr_date DATETIME;
    DECLARE last_name CHAR(20);

    SELECT lname INTO last_name FROM user;
    SELECT now() INTO curr_date;
    INSERT INTO logging VALUES
        (null, 'Update', 'reservation', last_name, curr_date);
END$
DELIMITER ;

DROP TRIGGER IF EXISTS logDRes;
DELIMITER $
CREATE TRIGGER logDRes AFTER DELETE ON reservation
FOR EACH ROW
BEGIN
    DECLARE curr_date DATETIME;
```

```

DECLARE last_name CHAR(20);

SELECT lname INTO last_name FROM user;
SELECT now() INTO curr_date;
INSERT INTO logging VALUES
(null, 'Delete', 'reservation', last_name, curr_date);
END$
DELIMITER ;

```

```

DROP TRIGGER IF EXISTS logIRes;
DELIMITER $
CREATE TRIGGER logIRes AFTER INSERT ON reservation
FOR EACH ROW
BEGIN
    DECLARE curr_date DATETIME;
    DECLARE last_name CHAR(20);

    SELECT lname INTO last_name FROM user;
    SELECT now() INTO curr_date;
    INSERT INTO logging VALUES
    (null, 'Insert', 'reservation', last_name, curr_date);
END$
DELIMITER ;

```

```

DROP TRIGGER IF EXISTS logUEv;
DELIMITER $
CREATE TRIGGER logUEv AFTER UPDATE ON event
FOR EACH ROW
BEGIN
    DECLARE curr_date DATETIME;
    DECLARE last_name CHAR(20);

    SELECT lname INTO last_name FROM user;
    SELECT now() INTO curr_date;
    INSERT INTO logging VALUES
    (null, 'Update', 'event', last_name, curr_date);
END$
DELIMITER ;

```

```

DROP TRIGGER IF EXISTS logDEv;
DELIMITER $
CREATE TRIGGER logDEv AFTER DELETE ON event
FOR EACH ROW
BEGIN
    DECLARE curr_date DATETIME;
    DECLARE last_name CHAR(20);

    SELECT lname INTO last_name FROM user;
    SELECT now() INTO curr_date;
    INSERT INTO logging VALUES
    (null, 'Delete', 'event', last_name, curr_date);
END$
DELIMITER ;

```

```

DROP TRIGGER IF EXISTS logIEv;
DELIMITER $
CREATE TRIGGER logIEv AFTER INSERT ON event
FOR EACH ROW
BEGIN
    DECLARE curr_date DATETIME;
    DECLARE last_name CHAR(20);

    SELECT lname INTO last_name FROM user;
    SELECT now() INTO curr_date;
    INSERT INTO logging VALUES
    (null, 'Insert', 'event', last_name, curr_date);
END$
DELIMITER ;

```

```

DROP TRIGGER IF EXISTS logUTravel;
DELIMITER $
CREATE TRIGGER logUTravel AFTER UPDATE ON travel_to
FOR EACH ROW

```

```

BEGIN
    DECLARE curr_date DATETIME;
    DECLARE last_name CHAR(20);

    SELECT lname INTO last_name FROM user;
    SELECT now() INTO curr_date;
    INSERT INTO logging VALUES
        (null, 'Update', 'travel_to', last_name, curr_date);
END$
DELIMITER ;

DROP TRIGGER IF EXISTS logDTravel;
DELIMITER $
CREATE TRIGGER logDTravel AFTER DELETE ON travel_to
FOR EACH ROW
BEGIN
    DECLARE curr_date DATETIME;
    DECLARE last_name CHAR(20);

    SELECT lname INTO last_name FROM user;
    SELECT now() INTO curr_date;
    INSERT INTO logging VALUES
        (null, 'Delete', 'travel_to', last_name, curr_date);
END$
DELIMITER ;

DROP TRIGGER IF EXISTS logITravel;
DELIMITER $
CREATE TRIGGER logITravel AFTER INSERT ON travel_to
FOR EACH ROW
BEGIN
    DECLARE curr_date DATETIME;
    DECLARE last_name CHAR(20);

    SELECT lname INTO last_name FROM user;
    SELECT now() INTO curr_date;
    INSERT INTO logging VALUES
        (null, 'Insert', 'travel_to', last_name, curr_date);
END$
DELIMITER ;

DROP TRIGGER IF EXISTS logUDest;
DELIMITER $
CREATE TRIGGER logUDest AFTER UPDATE ON destination
FOR EACH ROW
BEGIN
    DECLARE curr_date DATETIME;
    DECLARE last_name CHAR(20);

    SELECT lname INTO last_name FROM user;
    SELECT now() INTO curr_date;
    INSERT INTO logging VALUES
        (null, 'Update', 'destination', last_name, curr_date);
END$
DELIMITER ;

DROP TRIGGER IF EXISTS logDDest;
DELIMITER $
CREATE TRIGGER logDDest AFTER DELETE ON destination
FOR EACH ROW
BEGIN
    DECLARE curr_date DATETIME;
    DECLARE last_name CHAR(20);

    SELECT lname INTO last_name FROM user;
    SELECT now() INTO curr_date;
    INSERT INTO logging VALUES
        (null, 'Delete', 'destination', last_name, curr_date);
END$
DELIMITER ;

DROP TRIGGER IF EXISTS logIDest;
DELIMITER $

```

```

CREATE TRIGGER logIDest AFTER INSERT ON destination
FOR EACH ROW
BEGIN
    DECLARE curr_date DATETIME;
    DECLARE last_name CHAR(20);

    SELECT lname INTO last_name FROM user;
    SELECT now() INTO curr_date;
    INSERT INTO logging VALUES
    (null, 'Insert', 'destination', last_name, curr_date);
END$
DELIMITER ;

```

**Επεξήγηση Κώδικα:** Για να υλοποιήσουμε το ζητούμενο του ερωτήματος, δημιουργήσαμε 15 triggers. Δηλαδή 3 triggers για κάθε έναν από τους πίνακες που αναφέρονται στο ερώτημα. Πιο συγκεκριμένα, για κάθε πίνακα έχουμε ένα After Insert, ένα After Delete και ένα After Update trigger.

Στόχος των trigger είναι μετά από κάθε κίνηση σε οποιονδήποτε από τους ζητούμενους πίνακες, να ενημερώνεται ο πίνακας logging για την κίνηση αυτή.

Σε κάθε trigger από τους παραπάνω, παίρνουμε με δύο μεταβλητές τις τιμές curr\_date και last\_name, και έπειτα χρησιμοποιώντας τις, κάνουμε INSERT στον πίνακα logging με τις απαραίτητες πληροφορίες. Το last\_name που κρατάμε είναι το επώνυμο του it manager που έκανε κάθε φορά την αλλαγή και το παίρνουμε από έναν πίνακα user που έχουμε δημιουργήσει και κρατάμε κάθε φορά το επώνυμο του user που είναι συνδεδεμένος κάθε φορά στην εφαρμογή.

### Παράδειγμα Υλοποίησης:

Παράδειγμα υλοποίησης τους θα γίνει πιο κάτω στην ενότητα για το GUI καθώς η χρήση τους δεν γίνεται να φανεί μέσω της sql.

#### 3.1.4.2

##### Κώδικας:

```

DROP TRIGGER IF EXISTS no_change;
DELIMITER $

CREATE TRIGGER no_change BEFORE UPDATE ON trip
FOR EACH ROW
BEGIN
    DECLARE temp INT(11);

    IF (NEW.tr_departure<>OLD.tr_departure OR NEW.tr_return<>OLD.tr_return OR NEW.tr_cost<>OLD.tr_cost)
    THEN
        SELECT res_tr_id INTO temp FROM reservation
        WHERE res_tr_id = OLD.tr_id LIMIT 0,1;

        IF (temp IS NOT NULL)
        THEN
            SIGNAL SQLSTATE '45000'
            SET MESSAGE_TEXT = 'You cannot update this trip because there is already a registration.';
        END IF;
    END IF;
END$
DELIMITER ;

```

### Επεξήγηση Κώδικα:

Το συγκεκριμένο trigger ενεργοποιείται πριν κάθε Update στον πίνακα trip. Δουλειά του είναι να ελέγχει αν έχουν γίνει κρατήσεις σε αυτό το trip, αν έχουν γίνει και το Update που πάνε να κάνουμε είναι πάνω σε ημερομηνία ή στο κόστος, τότε δεν μας αφήνει να το κάνουμε.

Για να υλοποιήσουμε το ζητούμενο, ελέγχουμε ως παράμετρο στο if, αν κάποια από τις μεταβλητές που δεν θέλουμε να αλλάξει, πάει να αλλάξει με το Update. Αν γίνεται κάτι τέτοιο, τότε περνάμε μέσα στο If, και με ένα Select κοιτάμε αν υπάρχουν reservations για το συγκεκριμένο ταξίδι. Αν υπάρχουν τότε ακυρώνουμε την ενημέρωση και βγάζουμε μήνυμα λάθους.



### Παράδειγμα Υλοποίησης:

- 1) Πηγαίνοντας να αλλάξουμε την ημερομηνία αναχώρησης σε ένα ταξίδι που έχει ήδη reservations, παίρνουμε το παρακάτω μήνυμα:

```
UPDATE trip SET tr_departure='2023-01-30 11:00:00' WHERE tr_id=1;
Error Code: 1644. You cannot update this trip because there is already a registration.
```

- 2) Πηγαίνοντας να αλλάξουμε την ημερομηνία επιστροφής σε ένα ταξίδι που έχει ήδη reservations, παίρνουμε το παρακάτω μήνυμα:

```
UPDATE trip SET tr_return='2023-01-30 11:00:00' WHERE tr_id=1;
Error Code: 1644. You cannot update this trip because there is already a registration.
```

- 3) Πηγαίνοντας να αλλάξουμε το κόστος σε ένα ταξίδι που έχει ήδη reservations, παίρνουμε το παρακάτω μήνυμα:

```
UPDATE trip SET tr_cost=100.00 WHERE tr_id=1;
Error Code: 1644. You cannot update this trip because there is already a registration.
```

- 4) Πηγαίνοντας να αλλάξουμε όμως την ημερομηνία αναχώρησης σε ένα ταξίδι που δεν έχει reservations:

```
UPDATE trip SET tr_departure='2023-01-30 11:00:00' WHERE tr_id=21;
SELECT tr_departure FROM trip WHERE tr_id=21;
```

tr_departure
2023-01-30 11:00:00

Παρατηρούμε πως σε αυτήν την περίπτωση μας αφήνει να αλλάξουμε την ημερομηνία αναχώρησης καθώς σε αυτό το trip δεν έχει γίνει κανένα reservation ακόμα.

### 3.1.4.3

#### Κώδικας:

```
DROP TRIGGER IF EXISTS no_decrease;
DELIMITER $

CREATE TRIGGER no_decrease BEFORE UPDATE ON worker
FOR EACH ROW
BEGIN
    IF (NEW.wrk_salary < OLD.wrk_salary)
    THEN
        SIGNAL SQLSTATE '45000'
        SET MESSAGE_TEXT = 'You cannot decrease the salary of a worker';
    END IF;
END$
DELIMITER ;
```

**Επεξήγηση Κώδικα:** Αυτό το Trigger ενεργοποιείται κάθε φορά πριν γίνει κάποια ενημέρωση στον πίνακα worker. Ως παράμετρο σε ένα If Statement ελέγχουμε αν το update γίνεται πάνω στον μισθό του υπαλλήλου. Αν γίνεται εκεί, τότε μέσα στο If Statement ελέγχουμε αν το update έχει στόχο να μειώσει ή να αυξήσει τον μισθό. Αν προσπαθεί να τον μειώσει, τότε βγάζουμε μήνυμα λάθους και απαγορεύουμε την αλλαγή, αν προσπαθεί να τον αυξήσει τότε προχωράμε με το Update.

### Παράδειγμα Υλοποίησης:

Ας δούμε πρώτα την περίπτωση που αυξάνουμε τον μισθό ενός worker. Πριν κάνουμε το update ίσχυε το παρακάτω:

wrk_AT	wrk_salary
AM71514316	568.10

Καλούμε την συνάρτηση και έπειτα:

```
UPDATE worker SET wrk_salary=5000 WHERE wrk_AT='AM71514316';
```

wrk_AT	wrk_salary
AM71514316	5000.00

Παρατηρούμε πως σε αυτήν την περίπτωση όντως άλλαξε ο μισθός του worker.

Ας δούμε τώρα τι θα γίνει άμα προσπαθήσουμε να μειώσουμε τον μισθό του ίδιου worker.

```
UPDATE worker SET wrk_salary=10 WHERE wrk_AT='AM71514316'; Error Code: 1644. You cannot decrease the salary of a worker
```

## Έξτρα Triggers:

- **Κώδικας:**

```
DROP TRIGGER IF EXISTS capacity;
DELIMITER $
CREATE TRIGGER capacity BEFORE INSERT ON reservation
FOR EACH ROW
BEGIN
    DECLARE max_s TINYINT;
    CALL seats_trip(NEW.res_tr_id, @res);
    IF(@res=0)
        THEN SIGNAL SQLSTATE VALUE '45000'
        SET MESSAGE_TEXT = 'The maximum amount of seats have been Reached.';
    END IF;

    SELECT tr_maxseats INTO max_s FROM trip
    WHERE tr_id=NEW.res_tr_id;

    IF(NEW.res_seatnum>max_s)
        THEN SIGNAL SQLSTATE VALUE '45000'
        SET MESSAGE_TEXT = 'The seat you are trying to book exceeds the seat number limit of the bus.';
    END IF;
END$
DELIMITER ;
```

**Επεξήγηση Κώδικα:** Δημιουργήσαμε έναν trigger ο οποίος να αξιοποιεί τα δεδομένα που παίρνουμε από την Stored Procedure 'seats\_trip'. Πιο συγκεκριμένα, στόχος μας ήταν το trigger να μην επιτρέπει το reservation σε κάποιο trip το οποίο έχει ήδη φτάσει τον μέγιστο αριθμό κρατήσεων, και ταυτόχρονα να μην επιτρέπει σε κάποιον να κάνει κράτηση θέσης με αριθμό μεγαλύτερο από τον μέγιστο. Δηλαδή αν το ταξίδι έχει 50 θέσεις, να μην μπορέσει κάποιος να κάνει κράτηση στην θέση 51, καθώς αυτή δεν υπάρχει. Έτσι δημιουργούμε έναν trigger για BEFORE INSERT στον πίνακα reservation.

Για να πετύχουμε τα παραπάνω θα δημιουργήσουμε δύο If Statements.

Στο πρώτο, παίρνοντας την out τιμή από το Stored Procedure seats\_trip, ελέγχουμε ποια είναι η τιμή του. Αν η τιμή του είναι 0, σημαίνει πως δεν υπάρχουν άλλες διαθέσιμες θέσεις στο ταξίδι και έτσι δεν μπορεί να γίνει άλλο reservation σε αυτό. Έτσι βγάζει μήνυμα λάθους και δεν επιτρέπει την πραγματοποίηση του Insert.

Στο δεύτερο If Statement, ελέγχουμε αν η θέση που αντιστοιχεί στην κράτηση είναι μεγαλύτερη από την μεγαλύτερη θέση του ταξιδιού ή αν είναι η θέση 0. Αν ισχύει κάτι από τα δύο, βγαίνει πάλι μήνυμα λάθους και δεν επιτρέπεται να γίνει το συγκεκριμένο reservation.

### Παράδειγμα Υλοποίησης:

Για το πρώτο If Statement:

```
INSERT INTO reservation VALUES
(2, 22, 'Giannis', 'Papadopoulos', 'ADULT');
```

Error Code: 1644. The maximum amount of seats have been Reached.

Για να ελέγξουμε αν δουλεύει το δεύτερο If Statement, θα πάμε σε ένα ταξίδι που έχει ακόμα διαθέσιμες θέσεις. Σε αυτό το παράδειγμα θα προσπαθήσουμε να κάνουμε Insert στο ταξίδι με id=1 και να κρατήσουμε την θέση 21, ενώ το ταξίδι αυτό προσφέρει 20 θέσεις.

```
INSERT INTO reservation VALUES
(1, 21, 'Giannis', 'Papadopoulos', 'ADULT');
```

Error Code: 1644. The seat you are trying to book exceeds the seat number limit of the bus.

- **Κώδικας:**

```
DROP TRIGGER IF EXISTS check_driver;
DELIMITER $
CREATE TRIGGER check_driver BEFORE INSERT ON driver
FOR EACH ROW
BEGIN
    DECLARE temp1 char(10);
    DECLARE temp2 char(10);
    DECLARE temp3 char(10);

    SELECT adm_AT INTO temp1 FROM admin
    WHERE adm_AT = NEW.drv_AT
    limit 0,1;

    SELECT gui_AT INTO temp2 FROM guide
    WHERE gui_AT = NEW.drv_AT
    LIMIT 0,1;

    SELECT wrk_it_AT INTO temp3 FROM it
    WHERE wrk_it_AT = NEW.drv_AT
    LIMIT 0,1;

    IF(temp1 = NEW.drv_AT)
    THEN
        SIGNAL SQLSTATE '45000'
        SET MESSAGE_TEXT = 'You cannot add this worker as a driver because he is an Admin.';
    END IF;
    IF(temp2 = NEW.drv_AT)
    THEN
        SIGNAL SQLSTATE '45000'
        SET MESSAGE_TEXT = 'You cannot add this worker as a driver because he is a Guide.';
    END IF;
    IF(temp3 = NEW.drv_AT)
    THEN
        SIGNAL SQLSTATE '45000'
        SET MESSAGE_TEXT = 'You cannot add this worker as a driver because he is an IT Manager.';
    END IF;
END$
DELIMITER ;
```

```
DROP TRIGGER IF EXISTS check_admin;
DELIMITER $
CREATE TRIGGER check_admin BEFORE INSERT ON admin
FOR EACH ROW
BEGIN
    DECLARE temp1 char(10);
    DECLARE temp2 char(10);
    DECLARE temp3 char(10);

    SELECT drv_AT INTO temp1 FROM driver
    WHERE drv_AT = NEW.adm_AT
    limit 0,1;

    SELECT gui_AT INTO temp2 FROM guide
    WHERE gui_AT = NEW.adm_AT
    LIMIT 0,1;

    SELECT wrk_it_AT INTO temp3 FROM it
    WHERE wrk_it_AT = NEW.adm_AT
    LIMIT 0,1;

    IF(temp1 = NEW.adm_AT)
    THEN
        SIGNAL SQLSTATE '45000'
        SET MESSAGE_TEXT = 'You cannot add this worker as an admin because he is a Driver.';
    END IF;
    IF(temp2 = NEW.adm_AT)
    THEN
        SIGNAL SQLSTATE '45000'
        SET MESSAGE_TEXT = 'You cannot add this worker as an admin because he is a Guide.';
    END IF;
    IF(temp3 = NEW.adm_AT)
```

```

THEN
    SIGNAL SQLSTATE '45000'
    SET MESSAGE_TEXT = 'You cannot add this worker as an admin because he is an IT Manager.';
END IF;
END$
DELIMITER ;

```

```

DROP TRIGGER IF EXISTS check_guide;
DELIMITER $
CREATE TRIGGER check_guide BEFORE INSERT ON guide
FOR EACH ROW
BEGIN
    DECLARE temp1 char(10);
    DECLARE temp2 char(10);
    DECLARE temp3 char(10);

    SELECT drv_AT INTO temp1 FROM driver
    WHERE drv_AT = NEW.gui_AT
    limit 0,1;

    SELECT adm_AT INTO temp2 FROM admin
    WHERE adm_AT = NEW.gui_AT
    LIMIT 0,1;

    SELECT wrk_it_AT INTO temp3 FROM it
    WHERE wrk_it_AT = NEW.gui_AT
    LIMIT 0,1;

    IF(temp1 = NEW.gui_AT)
    THEN
        SIGNAL SQLSTATE '45000'
        SET MESSAGE_TEXT = 'You cannot add this worker as a Guide because he is a Driver.';
    END IF;
    IF(temp2 = NEW.gui_AT)
    THEN
        SIGNAL SQLSTATE '45000'
        SET MESSAGE_TEXT = 'You cannot add this worker as a Guide because he is an Admin.';
    END IF;
    IF(temp3 = NEW.gui_AT)
    THEN
        SIGNAL SQLSTATE '45000'
        SET MESSAGE_TEXT = 'You cannot add this worker asa Guide because he is an IT Manager.';
    END IF;
END$
DELIMITER ;

```

```

DROP TRIGGER IF EXISTS check_it;
DELIMITER $
CREATE TRIGGER check_it BEFORE INSERT ON it
FOR EACH ROW
BEGIN
    DECLARE temp1 char(10);
    DECLARE temp2 char(10);
    DECLARE temp3 char(10);

    SELECT drv_AT INTO temp1 FROM driver
    WHERE drv_AT = NEW.wrk_it_AT
    limit 0,1;

    SELECT adm_AT INTO temp2 FROM admin
    WHERE adm_AT = NEW.wrk_it_AT
    LIMIT 0,1;

    SELECT gui_AT INTO temp3 FROM guide
    WHERE gui_AT = NEW.wrk_it_AT
    LIMIT 0,1;

    IF(temp1 = NEW.wrk_it_AT)
    THEN
        SIGNAL SQLSTATE '45000'
        SET MESSAGE_TEXT = 'You cannot add this worker as an IT Manager because he is a Driver.';
    END IF;
END$
DELIMITER ;

```

```

END IF;
IF(temp2 = NEW.wrk_it_AT)
THEN
    SIGNAL SQLSTATE '45000'
    SET MESSAGE_TEXT = 'You cannot add this worker as an IT Manager because he is an Admin.';
END IF;
IF(temp3 = NEW.wrk_it_AT)
THEN
    SIGNAL SQLSTATE '45000'
    SET MESSAGE_TEXT = 'You cannot add this worker as an IT Manager because he is a Guide.';
END IF;
END$
DELIMITER ;

```

**Επεξήγηση Κώδικα:** Κάνοντας κάποια δοκιμαστικά Inserts για να δούμε την λειτουργία της βάσης μας, παρατηρήσαμε ότι ένας Worker μπορεί να εισαχθεί σε πάνω από μία υποκατηγορία. Δηλαδή για παράδειγμα να είναι και admin και driver ταυτόχρονα. Αυτό είναι κάτι που δεν θέλουμε να συμβαίνει. Για αυτόν τον λόγο αποφασίσαμε να δημιουργήσουμε ένα trigger για κάθε table που ανήκει στην κατηγορία worker. Δηλαδή για τους admin, guide, driver και it. Κάθε φορά πριν από insert σε έναν από αυτούς τους πίνακες θα ελέγχεται αν ο worker που πάμε να προσθέσουμε ανήκει ήδη κάπου αλλού ή όχι. Αν ανήκει κάπου αλλού θα απαγορεύουμε την εισαγωγή του στον πίνακα.

Πιο συγκεκριμένα, αυτό υλοποιείται δημιουργώντας για κάθε trigger, τρεις temp μεταβλητές. Για κάθε μία από αυτές κάνουμε ένα Select Into. Σε αυτό τσεκάρουμε αν υπάρχει το ίδιο AT με αυτό που πάμε να προσθέσουμε σε κάποιον από τους υπόλοιπους 3 πίνακες, αν δεν υπάρχει τότε η αντίστοιχη μεταβλητή παίρνει την τιμή null. Αν είναι και οι 3 μεταβλητές null, τότε επιτρέπεται η εισαγωγή στον πίνακα, αν έστω και μία από αυτές έχει τιμή, τότε απαγορεύουμε την εισαγωγή στον πίνακα και βγάζουμε κατάλληλο μήνυμα.

### Παράδειγμα Υλοποίησης:

Και τα 4 triggers στα οποία αναφερθήκαμε παραπάνω ακολουθούν το ίδιο ακριβώς μοτίβο, οπότε φτάνει να τεστάρουμε ένα για να δούμε ότι δουλεύει.

Ας δοκιμάσουμε λοιπόν να προσθέσουμε στον πίνακα driver, έναν worker που ανήκει ήδη στον πίνακα it.

wrk_it_AT	IT_AT	pass	start_date	end_date
AS85953443	IT01	pass0	2022-01-01 09:00:00	2022-12-30 09:00:00
AX57737772	IT02	pass1	2023-01-01 09:00:00	NULL
SA98242556	IT03	pass2	2022-01-02 09:00:00	NULL

Error Code: 1644. You cannot add this worker as a driver because he is an IT Manager.

Όπως βλέπουμε, βγαίνει μήνυμα λάθους και δεν μας επιτρέπεται η εισαγωγή αυτού του worker ως driver.

Ας σιγουρευτούμε τώρα ότι ο trigger δεν μας δημιουργεί πρόβλημα όταν πάμε να προσθέσουμε έναν driver χωρίς να υπάρχει ήδη αυτός ο worker σε κάποιους από τους πίνακες admin,it και guide.

```

SELECT * FROM driver;
INSERT INTO driver VALUES
('SA9824TEST', 'C', 'ABROAD',15);
SELECT * FROM driver;

```

drv_AT	drv_license	drv_route	drv_experience
AS33050797	D	LOCAL	28
AT53041686	B	ABROAD	7
QW82328598	C	ABROAD	23
SA88164961	C	ABROAD	12
SC32758869	D	LOCAL	1
SR77541159	B	ABROAD	23
SS46333586	B	ABROAD	23
SY94496257	C	LOCAL	32
TS73756168	D	ABROAD	15
WA59455740	B	LOCAL	11
WQ64668093	B	ABROAD	23
XR13453928	D	LOCAL	43
YR41149523	C	ABROAD	23
YT13536905	D	LOCAL	2

drv_AT	drv_license	drv_route	drv_experience
AS33050797	D	LOCAL	28
AT53041686	B	ABROAD	7
QW82328598	C	ABROAD	23
SA88164961	C	ABROAD	12
SA9824TEST	C	ABROAD	15
SC32758869	D	LOCAL	1
SR77541159	B	ABROAD	23
SS46333586	B	ABROAD	23
SY94496257	C	LOCAL	32
TS73756168	D	ABROAD	15
WA59455740	B	LOCAL	11
WQ64668093	B	ABROAD	23
XR13453928	D	LOCAL	43
YR41149523	C	ABROAD	23
YT13536905	D	LOCAL	2

Όπως βλέπουμε ο driver προστέθηκε με επιτυχία.

# Κεφάλαιο 4

## Εισαγωγή

Για την υλοποίηση του γραφικού περιβάλλοντος διαχείρισης της βάσης δεδομένων, αφότου ολοκληρώσαμε το σχεδιασμό του με τα βασικά δομικά στοιχεία που προσφέρει το Netbeans στην Java Swing, αποφασίσαμε πως θα θέλαμε η εφαρμογή να είναι πιο οπτικά ελκυστική στον χρήστη. Γι' αυτό τον λόγο, παρακολουθήσαμε κάποια Tutorial Βίντεο στο Youtube και χρησιμοποιήσαμε κάποια έτοιμα Plugins (παραθέτουμε τα links στην βιβλιογραφία). Έχοντας κατανοήσει πλήρως το περιεχόμενο των παραπάνω, ενσωματώσαμε στον κώδικά μας αυτά τα χαρακτηριστικά, ώστε να ταιριάζουν με την λειτουργικότητα της εφαρμογής μας. Επομένως, λόγω των παραπάνω και του μεγάλου όγκου κώδικα που χρειάζεται να περιγραφθεί για τις βασικές λειτουργίες της εφαρμογής, είναι συνετό να μην γίνει η πλήρης επεξήγηση αυτών των κομματιών κώδικα.

Πιο συγκεκριμένα, μερικά από τα Plugins που χρησιμοποιήσαμε ήταν τα, jCalendar-1.4 (για την επιλογή ημερομηνίας), swing-time-picker (για την επιλογή της ώρας).

## Τεκμηρίωση και Επεξήγηση

**!Προσοχή!** Στα παρακάτω κομμάτια κώδικα υπάρχουν και έξτρα λειτουργίες που θα αναλύσουμε στο κεφάλαιο 5

### login.java

#### Κώδικας:

```
private void loginActionPerformed(java.awt.event.ActionEvent evt) {
    try{
        Class.forName("com.mysql.cj.jdbc.Driver");
        Connection con = DriverManager.getConnection("jdbc:mysql://localhost:3306/travel_agency?zeroDateTimeBehavior=CONVERT_TO_NULL", "root", "root");
        String username = Username.getText();
        String password = Password.getText();

        Statement stm1 = con.createStatement();
        Statement stm2 = con.createStatement();
        String delete = "DELETE FROM user";
        PreparedStatement del = con.prepareStatement(delete);
        String logini = "SELECT * FROM it INNER JOIN worker ON wrk_it_AT = wrk_AT WHERE IT_AT='"+username+"' AND pass='"+password+"' AND end_date IS NULL";
        String loginm = "SELECT * FROM manages INNER JOIN worker ON mng_adm_AT = wrk_AT WHERE mng_username='"+username+"' AND mng_pass='"+password+"'";
        del.executeUpdate();
        ResultSet rs1 = stm1.executeQuery(logini);
        ResultSet rs2 = stm2.executeQuery(loginm);
        if(rs1.next()){
            lname = rs1.getString("wrk_lname");
            dispose();
            Main hpage1 = new Main();
            hpage1.show();
        }else if(rs2.next()){
            lname = rs2.getString("wrk_lname");
            branch = rs2.getInt("mng_br_code");
            dispose();
            Main_Mng hpage1 = new Main_Mng();
            hpage1.show();
        }else{
            JOptionPane.showMessageDialog(this, "Username or Password was Incorrect");
            Username.setText("");
            Password.setText("");
        }
        if(!"".equals(lname)){
            String insert = "INSERT INTO user VALUES('"+lname+"')";
            PreparedStatement ins = con.prepareStatement(insert);
            ins.executeUpdate();
        }
        con.close();
    }catch (HeadlessException | ClassNotFoundException | SQLException e){JOptionPane.showMessageDialog(this, e.getMessage());}
}
```

## Επεξήγηση:

Αρχικοποιούμε την σύνδεσή μας με την βάση δεδομένων, με ένα try-catch block, καθώς θέλουμε να πάρουμε τα δεδομένα του πίνακα IT, στον οποίο βρίσκουμε το username και password του IT Manager. Το String που παίρνει η συνάρτηση executeQuery σαν όρισμα (για την αρχικοποίηση του ResultSet), φαίνεται παρακάτω:

```
String logini = "SELECT * FROM it INNER JOIN worker ON wrk_it_AT = wrk_AT WHERE IT_AT='"+username+"' AND pass='"+password+"' AND end_date IS NULL";
```

Όπως παρατηρούμε θέτουμε όπου IT\_AT ίσο με το String 'username' και όπου pass ίσο με το String 'password' καθώς θέλουμε να ψάξουμε τον πίνακα IT για εγγραφές με αυτά τα ορίσματα.

Έπειτα, ελέγχουμε αν το ResultSet έχει εγγραφές (δηλαδή ο χρήστης δεν έχει εισάγει στοιχεία που δεν υπάρχουν ή είναι λανθασμένα) και θέτουμε στην μεταβλητή lname (Last Name ενός worker), την τιμή wrk\_lname που αντιστοιχεί στο επώνυμο του IT Manager που συνδέεται. Τέλος, εμφανίζεται το καινούριο JFrame με το κεντρικό μενού του IT Manager και το παρών JFrame κλείνει.

Εάν κάποια από τα στοιχεία είναι λανθασμένο ή δεν υπάρχει με ένα JOptionPane εμφανίζεται μήνυμα λάθους.

Στην συνέχεια, εισάγουμε στον πίνακα user το Last Name του χρήστη που συνδέεται, ώστε να το χρησιμοποιήσουμε για τα logs.

## [Main.java](#)

### Κώδικας:

```
public Main() {
    initComponents();
    setBackground(new Color(0, 0, 0, 0));
    menu.initMoving(Main.this);
    menu.addEventMenuSelected(new EventMenuSelected(){
        @Override
        public void selected(int index) {
            switch (index) {
                case 0:
                    setForm(new Form_Home());
                    break;
                case 1:
                    Insert_Selection select = new Insert_Selection();
                    select.show();
                    dispose();
                    break;
                case 2:
                    CheckTrip checktr = new CheckTrip();
                    checktr.show();
                    dispose();
                    break;
                case 3:
                    CheckOffers checkoff = new CheckOffers();
                    checkoff.show();
                    dispose();
                    break;
                case 4:
                    BranchInfo brinfo = new BranchInfo();
                    brinfo.show();
                    dispose();
                    break;
                case 5:
                    Employees empl = new Employees();
                    empl.show();
                    dispose();
                    break;
                case 8:
                    Logs log = new Logs();
                    log.show();
                    dispose();
                    break;
                case 9:
                    Login login = new Login();
                    login.show();
                    dispose();
                    break;
            }
        }
    });
    setForm(new Form_Home());
}
```

```
public void setForm(JComponent com){
    mainPanel.removeAll();
    mainPanel.add(com);
    mainPanel.repaint();
    mainPanel.revalidate();
}
```

### Επεξήγηση:

Το συγκεκριμένο JFrame είναι το Menu της εφαρμογής, δεν υλοποιεί το ίδιο κάποιο από τα ερωτήματα της εκφώνησης αλλά αποτελεί ένα dashboard με το οποίο ο IT Manager μπορεί να διαλέξει ποια ενέργεια θέλει να πραγματοποιήσει.

Περιέχει την συνάρτηση selected η οποία χρησιμοποιείται για την αλλαγή των jFrames ανάλογα με την επιλογή του IT Manager. Έτσι με ένα switch-case statement ανάλογα με το index της επιλογής του (που αλλάζει με βάση το ποιο JLabel προσπαθεί να επιλέξει από το μενού), ο IT Manager, εκτελεί τις λειτουργίες που υπάρχουν διαθέσιμες.

Ταυτόχρονα με την βοήθεια της συνάρτησης setForm (η οποία αρχικοποιεί το JPanel που υπάρχει στο JFrame 'Main'), εμφανίζεται το κύριο JPanel, το οποίο περιέχει βασικές πληροφορίες που ο IT Manager βλέπει καθώς εισέρχεται στην εφαρμογή.

### [Insert Selection.java](#)

### Κώδικας:

```
private void Selection1ActionPerformed(java.awt.event.ActionEvent evt) {
    String select = Selection1.getSelectedItem().toString();
    switch(select){
        case "branch":
            dispose();
            branch inspage1 = new branch();
            inspage1.show();
            break;
        case "phones":
            dispose();
            phones inspage2 = new phones();
            inspage2.show();
            break;
        case "worker":
            dispose();
            wrk inspage3 = new wrk();
            inspage3.show();
            break;
        case "it":
            dispose();
            it inspage4 = new it();
            inspage4.show();
            break;
        case "driver":
            dispose();
            drv inspage5 = new drv();
            inspage5.show();
            break;
        case "admin":
            dispose();
            adm inspage6 = new adm();
            inspage6.show();
            break;
        case "manages":
            dispose();
            mngs inspage7 = new mngs();
            inspage7.show();
            break;
        case "guide":
            dispose();
            gui inspage8 = new gui();
            inspage8.show();
            break;
        case "languages":
            dispose();
            lang inspage9 = new lang();
            inspage9.show();
            break;
        case "trip":
            dispose();
```



```

        trip inspage10 = new trip();
inspage10.show();
break;
case "event":
    dispose();
    evt inspage11 = new evt();
    inspage11.show();
    break;
case "destination":
    dispose();
    dst inspage12 = new dst();
    inspage12.show();
    break;
case "travel_to":
    dispose();
    trvl inspage13 = new trvl();
    inspage13.show();
    break;
case "reservation":
    dispose();
    rsv inspage14 = new rsv();
    inspage14.show();
    break;
case "offers":
    dispose();
    off inspage15 = new off();
    inspage15.show();
    break;
case "reservation_offers":
    dispose();
    ResOff inspage16 = new ResOff();
    inspage16.show();
    break;
    }
}

private void formWindowOpened(java.awt.event.WindowEvent evt) {
try{
    Class.forName("com.mysql.cj.jdbc.Driver");
    Connection con = DriverManager.getConnection("jdbc:mysql://localhost:3306/travel_agency?zeroDateTimeBehavior=CONVERT_TO_NULL", "root", "root");
    Statement stm = con.createStatement();
    String[] types = {"TABLE"};
    DatabaseMetaData metaData = con.getMetaData();
    ResultSet tables = metaData.getTables(null, null, "%", types);
    DefaultComboBoxModel mod = new DefaultComboBoxModel();
    mod.removeAllElements();
    while (tables.next()){
        if(!"sys_config".equals(tables.getString("TABLE_NAME")) && !"logging".equals(tables.getString("TABLE_NAME")) && !"user".equals(tables.getString("TABLE_NAME"))
            && !"new".equals(tables.getString("TABLE_NAME")) && !"final_paid".equals(tables.getString("TABLE_NAME")) &&
!"new_res_off".equals(tables.getString("TABLE_NAME"))){
            mod.addElement(tables.getString("TABLE_NAME"));
        }
    }
    Selection1.setModel(mod);
    con.close();
}catch(ClassNotFoundException | SQLException e){OptionPane.showMessageDialog(this, e.getMessage());}
}

```

## Επεξήγηση:

Αυτό το JFrame χρησιμοποιείται στην περίπτωση που ο IT Manager θέλει να προβάλει ή να αλλάξει κάποιον πίνακα από την βάση δεδομένων. Επομένως, με το event formWindowOpened (που τρέχει με το άνοιγμα του συγκεκριμένου JFrame), αρχικοποιούμε μία σύνδεση με την βάση δεδομένων, με ένα try-catch block, για να έχουμε πρόσβαση στα MetaData της βάσης. Δημιουργούμε ένα custom JComboBox το οποίο παίρνει δυναμικά τα ονόματα των πινάκων της βάσης δεδομένων και δίνει στον χρήστη την δυνατότητα να επιλέξει ποιον από αυτούς θέλει να επεξεργαστεί. Έτσι, εάν προσθέσουμε κάποιον πίνακα στην βάση που θέλουμε να διαχειρίζεται από τον IT Manager θα εμφανίζεται σαν επιλογή χωρίς να πρέπει να το προσθέσουμε χειροκίνητα στο JComboBox. Τέλος, με ένα switch-case statement, ανάλογα με την επιλογή του IT Manager εμφανίζεται το κατάλληλο JFrame και κλείνει το παρόν JFrame.

# Για την επεξήγηση των παρακάτω αρχείων .java προχωρήστε [εδώ](#)

[trip.java](#)

## Κώδικας:

```
static Timestamp t1;
static Timestamp t2;
static Timestamp temp;
public void convertTimeStamp(){
    String time1 = PickTime1.getText();
    String time2 = PickTime2.getText();
    time1=time1.replace(" ", "");
    time2=time2.replace(" ", "");
    if(time1.contains("PM")){
        String[] parts = time1.split(":");
        String part1 = parts[0];
        String part2 = parts[1];
        if(!"12".equals(part1)){
            int part1int = Integer.parseInt(part1)+12;
            part1 = Integer.toString(part1int);
            time1 = part1.concat(":").concat(part2);
        }else time1 = part1.concat(":").concat(part2);
    }else if(time1.contains("AM")){
        String[] parts = time1.split(":");
        String part1 = parts[0];
        String part2 = parts[1];
        if("12".equals(part1)){
            part1 = "00";
            time1 = part1.concat(":").concat(part2);
        }else time1 = part1.concat(":").concat(part2);
    }else{
        String[] parts = time1.split(":");
        String part1 = parts[0];
        String part2 = parts[1];
        time1 = part1.concat(":").concat(part2);
    }
    time1=time1.replace("PM", "");
    time1=time1.replace("AM", "");

    if(time2.contains("PM")){
        String[] parts = time2.split(":");
        String part1 = parts[0];
        String part2 = parts[1];
        if(!"12".equals(part1)){
            int part1int = Integer.parseInt(part1)+12;
            part1 = Integer.toString(part1int);
            time2 = part1.concat(":").concat(part2);
        }else time2 = part1.concat(":").concat(part2);
    }else{
        String[] parts = time2.split(":");
        String part1 = parts[0];
        String part2 = parts[1];
        if("12".equals(part1)){
            part1 = "00";
            time2 = part1.concat(":").concat(part2);
        }else time2 = part1.concat(":").concat(part2);
    }
    time2=time2.replace("PM", "");
    time2=time2.replace("AM", "");
    time1=time1.concat(":00");
    time2=time2.concat(":00");
    DateFormat dateFormat1 = new SimpleDateFormat("yyyy-MM-dd ");
    String strDate1 = dateFormat1.format(jDateChooser1.getDate());
    String date1 = strDate1.concat(time1);
    DateFormat dateFormat2 = new SimpleDateFormat("yyyy-MM-dd ");
    String strDate2 = dateFormat2.format(jDateChooser2.getDate());
    String date2 = strDate2.concat(time2);
    t1 = java.sql.Timestamp.valueOf(date1);
    t2 = java.sql.Timestamp.valueOf(date2);
}

public void updateTable(){
    try{
        Class.forName("com.mysql.cj.jdbc.Driver");
        Connection con = DriverManager.getConnection("jdbc:mysql://localhost:3306/travel_agency?zeroDateTimeBehavior=CONVERT_TO_NULL", "root", "root");
```

```

String select="SELECT * FROM trip;";
Statement slct = con.createStatement();
ResultSet rs = slct.executeQuery(select);
DefaultTableModel tbModel= (DefaultTableModel) TripTable.getModel();
tbModel.setNumRows(0);
while(rs.next()){
    String id = rs.getString("tr_id");
    String departure = rs.getString("tr_departure");
    String ret = rs.getString("tr_return");
    String maxseats = rs.getString("tr_maxseats");
    String cost = rs.getString("tr_cost");
    String code = rs.getString("tr_br_code");
    String gui = rs.getString("tr_gui_AT");
    String drv = rs.getString("tr_drv_AT");
    String tb_data[]= {id, departure, ret, maxseats, cost, code, gui, drv};
    tbModel.addRow(tb_data);
}
con.close();
}catch(ClassNotFoundException | SQLException e){OptionPane.showMessageDialog(this, e.getMessage());}
}

public void updateCombo(){
    PickTime1.setText("Set Time");
    PickTime2.setText("Set Time");
    try{
        Class.forName("com.mysql.cj.jdbc.Driver");
        Connection con = DriverManager.getConnection("jdbc:mysql://localhost:3306/travel_agency?zeroDateTimeBehavior=CONVERT_TO_NULL", "root", "root");
        String select="SELECT br_code FROM branch;";
        Statement slct = con.createStatement();
        ResultSet rs = slct.executeQuery(select);
        DefaultComboBoxModel mod = new DefaultComboBoxModel();
        mod.removeAllElements();
        mod.addElement("");
        while(rs.next()){
            int box = rs.getInt("br_code");
            mod.addElement(box);
        }
        tr_br_code.setModel(mod);
        con.close();
    }catch(ClassNotFoundException | SQLException e){OptionPane.showMessageDialog(this, e.getMessage());}
}

private void PickTime2MouseClicked(java.awt.event.MouseEvent evt) {
    timePicker2.showPopup(this, 100, 100);
    String time = timePicker2.getSelectedTime();
    time=time.replace("MP", "PM");
    PickTime2.setText(time);
}

private void PickTime1MouseClicked(java.awt.event.MouseEvent evt) {
    timePicker1.showPopup(this, 100, 100);
    String time = timePicker1.getSelectedTime();
    time=time.replace("MP", "PM");
    PickTime1.setText(time);
}

private void formWindowOpened(java.awt.event.WindowEvent evt) {
    jLabel5.setVisible(false);
    jLabel8.setVisible(false);
    tr_gui_AT.setVisible(false);
    tr_drv_AT.setVisible(false);
    updateCombo();
    updateTable();
}

private void TripTableMouseClicked(java.awt.event.MouseEvent evt) {
    DefaultTableModel tbModel= (DefaultTableModel) TripTable.getModel();
    String dep = tbModel.getValueAt(TripTable.getSelectedRow(), 1).toString();
    String ret = tbModel.getValueAt(TripTable.getSelectedRow(), 2).toString();
    String max = tbModel.getValueAt(TripTable.getSelectedRow(), 3).toString();
    String cost = tbModel.getValueAt(TripTable.getSelectedRow(), 4).toString();
    String br_code = tbModel.getValueAt(TripTable.getSelectedRow(), 5).toString();
    String gui = tbModel.getValueAt(TripTable.getSelectedRow(), 6).toString();
    String drv = tbModel.getValueAt(TripTable.getSelectedRow(), 7).toString();
    tr_maxseats.setText(max);
    tr_cost.setText(cost);
    tr_br_code.setSelectedItem(Integer.valueOf(br_code));
    tr_gui_AT.setSelectedItem(gui);
    tr_drv_AT.setSelectedItem(drv);
}

```

```

String[] parts1 = dep.split(" ");
String part1s = parts1[0];
String part2s = parts1[1];

jDateChooser1.setDate(java.sql.Date.valueOf(part1s));
PickTime1.setText(part2s);

String[] parts2 = ret.split(" ");
String part1e = parts2[0];
String part2e = parts2[1];
jDateChooser2.setDate(java.sql.Date.valueOf(part1e));
PickTime2.setText(part2e);
}

private void InsertActionPerformed(java.awt.event.ActionEvent evt) {
    try{
        Class.forName("com.mysql.cj.jdbc.Driver");
        Connection con = DriverManager.getConnection("jdbc:mysql://localhost:3306/travel_agency?zeroDateTimeBehavior=CONVERT_TO_NULL", "root", "root");
        String insert="INSERT INTO trip(tr_id,tr_departure,tr_return,tr_maxseats,tr_cost,tr_br_code,tr_gui_AT,tr_drv_AT) VALUES(null,?,?,?,?);";
        PreparedStatement insrt = con.prepareStatement(insert);
        convertTimeStamp();
        insrt.setTimestamp(1,t1);
        insrt.setTimestamp(2,t2);
        insrt.setInt(3, Integer.parseInt(tr_maxseats.getText()));
        insrt.setFloat(4,Float.parseFloat(tr_cost.getText()));
        insrt.setInt(5, Integer.parseInt(tr_br_code.getSelectedItem().toString()));
        insrt.setString(6, tr_gui_AT.getSelectedItem().toString());
        insrt.setString(7, tr_drv_AT.getSelectedItem().toString());
        insrt.execute();
        updateTable();
        JOptionPane.showMessageDialog(this, "Inserted Succesfully");
    }catch(ClassNotFoundException | SQLException e){JOptionPane.showMessageDialog(this, e.getMessage());}
}

private void Delete1ActionPerformed(java.awt.event.ActionEvent evt) {
    try{
        Class.forName("com.mysql.cj.jdbc.Driver");
        Connection con = DriverManager.getConnection("jdbc:mysql://localhost:3306/travel_agency?zeroDateTimeBehavior=CONVERT_TO_NULL", "root", "root");
        String delete="DELETE FROM trip WHERE tr_id = ?";
        PreparedStatement dlt = con.prepareStatement(delete);
        DefaultTableModel tbModel= (DefaultTableModel) TripTable.getModel();
        dlt.setInt(1, Integer.parseInt(tbModel.getValueAt(TripTable.getSelectedRow(), 0).toString()));
        dlt.executeUpdate();
        updateTable();
        jDateChooser1.setDate(null);
        jDateChooser2.setDate(null);
        PickTime1.setText("");
        PickTime2.setText("");
        tr_maxseats.setText("");
        tr_cost.setText("");
        JOptionPane.showMessageDialog(this, "Deleted Succesfully");
        con.close();
    }catch(ClassNotFoundException | SQLException e){JOptionPane.showMessageDialog(this, e.getMessage());}
}

private void Update1ActionPerformed(java.awt.event.ActionEvent evt) {
    convertTimeStamp();
    try{
        Class.forName("com.mysql.cj.jdbc.Driver");
        Connection con = DriverManager.getConnection("jdbc:mysql://localhost:3306/travel_agency?zeroDateTimeBehavior=CONVERT_TO_NULL", "root", "root");
        String update="UPDATE trip SET tr_departure = ?, tr_return = ?, tr_maxseats = ?, tr_cost = ?, tr_br_code = ?, tr_gui_AT = ?, tr_drv_AT = ? WHERE tr_id = ?";
        PreparedStatement upd = con.prepareStatement(update);
        DefaultTableModel tbModel= (DefaultTableModel) TripTable.getModel();
        upd.setTimestamp(1, t1);
        upd.setTimestamp(2, t2);
        upd.setInt(3, Integer.parseInt(tr_maxseats.getText()));
        upd.setFloat(4, Float.parseFloat(tr_cost.getText()));
        upd.setInt(5, Integer.parseInt(tr_br_code.getSelectedItem().toString()));
        upd.setString(6, tr_gui_AT.getSelectedItem().toString());
        upd.setString(7, tr_drv_AT.getSelectedItem().toString());
        upd.setInt(8, Integer.parseInt(tbModel.getValueAt(TripTable.getSelectedRow(), 0).toString()));
        upd.executeUpdate();
        updateTable();
        JOptionPane.showMessageDialog(this, "Updated Succesfully");
        con.close();
    }catch(ClassNotFoundException | SQLException e){JOptionPane.showMessageDialog(this, e.getMessage());}
}

```

```

}

private void CancelActionPerformed(java.awt.event.ActionEvent evt) {
    dispose();
    Main main = new Main();
    main.show();
}

private void tr_br_codeActionPerformed(java.awt.event.ActionEvent evt) {
    try{
        Class.forName("com.mysql.cj.jdbc.Driver");
        Connection con = DriverManager.getConnection("jdbc:mysql://localhost:3306/travel_agency?zeroDateTimeBehavior=CONVERT_TO_NULL", "root", "root");
        DefaultComboBoxModel mod2 = new DefaultComboBoxModel();
        DefaultComboBoxModel mod3 = new DefaultComboBoxModel();
        if(!"".equals(tr_br_code.getSelectedItem().toString())){
            mod2.removeAllElements();
            mod3.removeAllElements();
            jLabel5.setVisible(true);
            jLabel8.setVisible(true);
            tr_gui_AT.setVisible(true);
            tr_drv_AT.setVisible(true);
            String where = tr_br_code.getSelectedItem().toString();
            String select2="SELECT DISTINCT gui_AT FROM guide INNER JOIN worker ON wrk_AT = gui_AT WHERE wrk_br_code = "+where+"";
            try{
                Statement slct2 = con.createStatement();
                ResultSet rs2 = slct2.executeQuery(select2);
                while(rs2.next()){
                    String box2 = rs2.getString("gui_AT");
                    mod2.addElement(box2);
                }
                tr_gui_AT.setModel(mod2);
                String select3="SELECT DISTINCT drv_AT FROM driver INNER JOIN worker ON wrk_AT = drv_AT WHERE wrk_br_code = "+where+"";
                Statement slct3 = con.createStatement();
                ResultSet rs3 = slct3.executeQuery(select3);
                while(rs3.next()){
                    String box3 = rs3.getString("drv_AT");
                    mod3.addElement(box3);
                }
                tr_drv_AT.setModel(mod3);

            }catch(SQLException c){System.out.println(c.getMessage());}
        }con.close();
    }catch(ClassNotFoundException | SQLException e){JOptionPane.showMessageDialog(this, e.getMessage());}
}

```

## ResOff.java

### Κώδικας:

```

public void updateTable(){
    try{
        Class.forName("com.mysql.cj.jdbc.Driver");
        Connection con = DriverManager.getConnection("jdbc:mysql://localhost:3306/travel_agency?zeroDateTimeBehavior=CONVERT_TO_NULL", "root", "root");
        String select="SELECT * FROM reservation_offers;";
        Statement slct = con.createStatement();
        ResultSet rs = slct.executeQuery(select);
        DefaultTableModel tbModel= (DefaultTableModel) ResOffTable.getModel();
        tbModel.setNumRows(0);
        while(rs.next()){
            String id = rs.getString("res_of_id");
            String name = rs.getString("first_name");
            String lname = rs.getString("last_name");
            String offid = rs.getString("off_id");
            String adv = Float.toString(rs.getFloat("adv_pay"));
            String tb_data[]= {id, name, lname, offid, adv};
            tbModel.addRow(tb_data);
        }
        con.close();
    }catch(ClassNotFoundException | SQLException e){JOptionPane.showMessageDialog(this, e.getMessage());}
}

public void updateCombo(){
    try{
        Class.forName("com.mysql.cj.jdbc.Driver");
        Connection con = DriverManager.getConnection("jdbc:mysql://localhost:3306/travel_agency?zeroDateTimeBehavior=CONVERT_TO_NULL", "root", "root");
        String select="SELECT off_code FROM offers;";
        Statement slct = con.createStatement();
        ResultSet rs = slct.executeQuery(select);
    }
}

```

```

        DefaultComboBoxModel mod = new DefaultComboBoxModel();
        mod.removeAllElements();
        while(rs.next()){
            String box = "Offer Code: "+rs.getInt("off_code")+"";
            mod.addElement(box);
        }
        off_id.setModel(mod);
        con.close();
    }catch(ClassNotFoundException | SQLException e){OptionPane.showMessageDialog(this, e.getMessage());}
}

private void formWindowOpened(java.awt.event.WindowEvent evt) {
    updateCombo();
    updateTable();
}

private void UpdateActionPerformed(java.awt.event.ActionEvent evt) {
    try{
        DefaultTableModel tbModel= (DefaultTableModel) ResOffTable.getModel();
        Class.forName("com.mysql.cj.jdbc.Driver");
        Connection con = DriverManager.getConnection("jdbc:mysql://localhost:3306/travel_agency?zeroDateTimeBehavior=CONVERT_TO_NULL", "root", "root");
        String update="UPDATE reservation_offers SET first_name = ?, last_name = ?, off_id = ?, adv_pay = ? WHERE res_of_id = ?";
        PreparedStatement upd = con.prepareStatement(update);
        String code = off_id.getSelectedItem().toString().replaceAll("[^0-9]", "");
        int i=Integer.parseInt(code);
        upd.setString(1, first_name.getText());
        upd.setString(2, last_name.getText());
        upd.setInt(3, i);
        upd.setFloat(4, Float.parseFloat(adv_pay.getText()));
        upd.setInt(5, Integer.parseInt(tbModel.getValueAt(ResOffTable.getSelectedRow(), 0).toString()));
        upd.executeUpdate();
        updateTable();
        JOptionPane.showMessageDialog(this, "Updated Succesfully");
        con.close();
    }catch(ClassNotFoundException | SQLException e){OptionPane.showMessageDialog(this, e.getMessage());}
}

private void InsertActionPerformed(java.awt.event.ActionEvent evt) {
    try{
        Class.forName("com.mysql.cj.jdbc.Driver");
        Connection con = DriverManager.getConnection("jdbc:mysql://localhost:3306/travel_agency?zeroDateTimeBehavior=CONVERT_TO_NULL", "root", "root");
        String insert="INSERT INTO reservation_offers(res_of_id,first_name,last_name,off_id,adv_pay) VALUES(null,?,?,?,?,?)";
        PreparedStatement insrt = con.prepareStatement(insert);
        String code = off_id.getSelectedItem().toString().replaceAll("[^0-9]", "");
        int i=Integer.parseInt(code);
        insrt.setString(1,first_name.getText());
        insrt.setString(2,last_name.getText());
        insrt.setInt(3,i);
        insrt.setFloat(4,Float.parseFloat(adv_pay.getText()));
        insrt.execute();
        updateTable();
        JOptionPane.showMessageDialog(this, "Inserted Succesfully");
    }catch(ClassNotFoundException | SQLException e){OptionPane.showMessageDialog(this, e.getMessage());}
}

private void CancelActionPerformed(java.awt.event.ActionEvent evt) {
    dispose();
    Main main = new Main();
    main.show();
}

private void DeleteActionPerformed(java.awt.event.ActionEvent evt) {
    try{
        DefaultTableModel tbModel= (DefaultTableModel) ResOffTable.getModel();
        Class.forName("com.mysql.cj.jdbc.Driver");
        Connection con = DriverManager.getConnection("jdbc:mysql://localhost:3306/travel_agency?zeroDateTimeBehavior=CONVERT_TO_NULL", "root", "root");
        String delete="DELETE FROM reservation_offers WHERE res_of_id = ?";
        PreparedStatement dlt = con.prepareStatement(delete);
        dlt.setInt(1, Integer.parseInt(tbModel.getValueAt(ResOffTable.getSelectedRow(), 0).toString()));
        dlt.executeUpdate();
        updateTable();
        first_name.setText("");
        last_name.setText("");
        adv_pay.setText("");
        JOptionPane.showMessageDialog(this, "Deleted Succesfully");
        con.close();
    }catch(ClassNotFoundException | SQLException e){OptionPane.showMessageDialog(this, e.getMessage());}
}

```

```
private void ResOffTableMouseClicked(java.awt.event.MouseEvent evt) {
    DefaultTableModel tbModel= (DefaultTableModel) ResOffTable.getModel();
    String name = tbModel.getValueAt(ResOffTable.getSelectedRow(), 1).toString();
    String lname = tbModel.getValueAt(ResOffTable.getSelectedRow(), 2).toString();
    String id = tbModel.getValueAt(ResOffTable.getSelectedRow(), 3).toString();
    String adv = tbModel.getValueAt(ResOffTable.getSelectedRow(), 4).toString();
    first_name.setText(name);
    last_name.setText(lname);
    off_id.setSelectedItem("Offer Code: "+id+"");
    adv_pay.setText(adv);
}
```

## adm.java

### Κώδικας:

```
public void updateTable(){
    try{
        Class.forName("com.mysql.cj.jdbc.Driver");
        Connection con = DriverManager.getConnection("jdbc:mysql://localhost:3306/travel_agency?zeroDateTimeBehavior=CONVERT_TO_NULL", "root", "root");
        String select="SELECT * FROM admin;";
        Statement slct = con.createStatement();
        ResultSet rs = slct.executeQuery(select);
        DefaultTableModel tbModel= (DefaultTableModel) AdminTable.getModel();
        tbModel.setNumRows(0);
        while(rs.next()){
            String at = rs.getString("adm_AT");
            String language = rs.getString("adm_type");
            String diploma = rs.getString("adm_diploma");
            String tb_data[] = {at,language,diploma};
            tbModel.addRow(tb_data);
        }
        con.close();
    }catch(ClassNotFoundException | SQLException e){JOptionPane.showMessageDialog(this, e.getMessage());}
}

public void updateCombo(){
    try{
        Class.forName("com.mysql.cj.jdbc.Driver");
        Connection con = DriverManager.getConnection("jdbc:mysql://localhost:3306/travel_agency?zeroDateTimeBehavior=CONVERT_TO_NULL", "root", "root");
        String select ="SELECT wrk_AT FROM worker;";
        Statement slct = con.createStatement();
        ResultSet rs = slct.executeQuery(select);
        DefaultComboBoxModel mod = new DefaultComboBoxModel();
        mod.removeAllElements();
        while(rs.next()){mod.addElement(rs.getString("wrk_AT"));}
        wrk_AT.setModel(mod);
        con.close();
    }catch(ClassNotFoundException | SQLException e){JOptionPane.showMessageDialog(this, e.getMessage());}
}

private void formWindowOpened(java.awt.event.WindowEvent evt) {
    updateCombo();
    updateTable();
}

private void AdminTableMouseClicked(java.awt.event.MouseEvent evt) {
    DefaultTableModel tbModel= (DefaultTableModel) AdminTable.getModel();
    String at = tbModel.getValueAt(AdminTable.getSelectedRow(), 0).toString();
    String type = tbModel.getValueAt(AdminTable.getSelectedRow(), 1).toString();
    String diploma = tbModel.getValueAt(AdminTable.getSelectedRow(), 2).toString();
    wrk_AT.setSelectedItem(at);
    adm_type.setSelectedItem(type);
    adm_diploma.setText(diploma);
}

private void InsertActionPerformed(java.awt.event.ActionEvent evt) {
    try{
        Class.forName("com.mysql.cj.jdbc.Driver");
        Connection con = DriverManager.getConnection("jdbc:mysql://localhost:3306/travel_agency?zeroDateTimeBehavior=CONVERT_TO_NULL", "root", "root");
        String insert="INSERT INTO admin(adm_AT,adm_diploma,adm_type) VALUES(?,?,?)";
        PreparedStatement insrt = con.prepareStatement(insert);
        insrt.setString(1,wrk_AT.getSelectedItem().toString());
        insrt.setString(2,adm_diploma.getText());
        insrt.setString(3,adm_type.getSelectedItem().toString());
        insrt.execute();
        updateTable();
        JOptionPane.showMessageDialog(this, "Inserted Succesfully");
    }
```

```

}catch(ClassNotFoundException | SQLException e){JOptionPane.showMessageDialog(this, e.getMessage());}
}

private void CancelActionPerformed(java.awt.event.ActionEvent evt) {
    dispose();
    Main main = new Main();
    main.show();
}

private void DeleteActionPerformed(java.awt.event.ActionEvent evt) {
    try{
        String fname = null;
        String lname = null;
        String test = null;
        String msg = "You cannot Delete this User. This User is an Administrative.";
        String msg2 = "This Worker is not an Admin!";
        Class.forName("com.mysql.cj.jdbc.Driver");
        Connection con = DriverManager.getConnection("jdbc:mysql://localhost:3306/travel_agency?zeroDateTimeBehavior=CONVERT_TO_NULL", "root", "root");
        String delete="DELETE FROM admin WHERE adm_AT = ?";
        String select1="SELECT wrk_name FROM worker INNER JOIN admin ON adm_AT=wrk_AT WHERE wrk_AT = ? AND adm_type='ADMINISTRATIVE'";
        String select2="SELECT wrk_lname FROM worker INNER JOIN admin ON adm_AT=wrk_AT WHERE wrk_AT = ? AND adm_type='ADMINISTRATIVE'";
        String select3="SELECT adm_AT FROM admin";
        PreparedStatement dlt = con.prepareStatement(delete);
        PreparedStatement slc1 = con.prepareStatement(select1);
        PreparedStatement slc2 = con.prepareStatement(select2);
        PreparedStatement slc3 = con.prepareStatement(select3);
        slc1.setString(1, wrk_AT.getSelectedItem().toString());
        slc2.setString(1, wrk_AT.getSelectedItem().toString());
        dlt.setString(1, wrk_AT.getSelectedItem().toString());
        ResultSet rs1 = slc1.executeQuery();
        ResultSet rs2 = slc2.executeQuery();
        ResultSet rs4 = slc3.executeQuery();
        if(rs1.next()==true){fname = rs1.getString("wrk_name");}
        if(rs2.next()==true){lname = rs2.getString("wrk_lname");}
        String s = "{CALL admin_check('"+fname+"', '"+lname+"')}";
        CallableStatement cs = con.prepareCall(s);
        cs.execute();
        ResultSet rs3 = cs.getResultSet();
        System.out.println(fname);
        System.out.println(lname);
        if(rs3 != null)rs3.next();
        if(rs3.getString(1).equals(msg)){
            JOptionPane.showMessageDialog(this, msg);
        }else if(rs3.getString(1).equals(msg2) && fname == null){
            while(rs4.next()){
                if(rs4.getString("adm_AT").equals(wrk_AT.getSelectedItem().toString())){
                    test = "found";
                    break;
                }else test = "not found";
            }if ("found".equals(test)){
                dlt.execute();
                JOptionPane.showMessageDialog(this, "Deleted Successfully");
            }else JOptionPane.showMessageDialog(this, msg2);
        }
        updateTable();
        con.close();
    }catch(ClassNotFoundException | SQLException e){JOptionPane.showMessageDialog(this, e.getMessage());}
}

private void UpdateActionPerformed(java.awt.event.ActionEvent evt) {
    try{
        DefaultTableModel tbModel= (DefaultTableModel) AdminTable.getModel();
        Class.forName("com.mysql.cj.jdbc.Driver");
        Connection con = DriverManager.getConnection("jdbc:mysql://localhost:3306/travel_agency?zeroDateTimeBehavior=CONVERT_TO_NULL", "root", "root");
        String update="UPDATE admin SET adm_type = ?, adm_diploma = ? WHERE adm_AT = ?";
        PreparedStatement upd = con.prepareStatement(update);
        upd.setString(1, adm_type.getSelectedItem().toString());
        upd.setString(2, adm_diploma.getText());
        upd.setString(3, tbModel.getValueAt(AdminTable.getSelectedRow(), 0).toString());
        upd.executeUpdate();
        updateTable();
        JOptionPane.showMessageDialog(this, "Updated Successfully");
        con.close();
    }catch(ClassNotFoundException | SQLException e){JOptionPane.showMessageDialog(this, e.getMessage());}
}

```



Κώδικας:

```
public void updateTable(){
    try{
        Class.forName("com.mysql.cj.jdbc.Driver");
        Connection con = DriverManager.getConnection("jdbc:mysql://localhost:3306/travel_agency?zeroDateTimeBehavior=CONVERT_TO_NULL", "root", "root");
        String select="SELECT * FROM branch;";
        Statement slct = con.createStatement();
        ResultSet rs = slct.executeQuery(select);
        DefaultTableModel tbModel= (DefaultTableModel) BranchTable.getModel();
        tbModel.setNumRows(0);
        while(rs.next()){
            String code = rs.getString("br_code");
            String street = rs.getString("br_street");
            String num = rs.getString("br_num");
            String city = rs.getString("br_city");
            String tb_data[]= {code, street, num, city};
            tbModel.addRow(tb_data);
        }
        con.close();
    }catch(ClassNotFoundException | SQLException e){OptionPane.showMessageDialog(this, e.getMessage());}
}

private void formWindowOpened(java.awt.event.WindowEvent evt) {
    updateTable();
}

private void BranchTableMouseClicked(java.awt.event.MouseEvent evt) {
    DefaultTableModel tbModel= (DefaultTableModel) BranchTable.getModel();
    String street = tbModel.getValueAt(BranchTable.getSelectedRow(), 1).toString();
    String num = tbModel.getValueAt(BranchTable.getSelectedRow(), 2).toString();
    String city = tbModel.getValueAt(BranchTable.getSelectedRow(), 3).toString();
    br_street.setText(street);
    br_num.setText(num);
    br_city.setText(city);
}

private void InsertActionPerformed(java.awt.event.ActionEvent evt) {
    try{
        Class.forName("com.mysql.cj.jdbc.Driver");
        Connection con = DriverManager.getConnection("jdbc:mysql://localhost:3306/travel_agency?zeroDateTimeBehavior=CONVERT_TO_NULL", "root", "root");
        String insert="INSERT INTO branch(br_code,br_street,br_num,br_city) VALUES(null,?,?,?)";
        PreparedStatement insrt = con.prepareStatement(insert);
        insrt.setString(1,br_street.getText());
        insrt.setInt(2,Integer.parseInt(br_num.getText()));
        insrt.setString(3,br_city.getText());
        insrt.execute();
        updateTable();
        JOptionPane.showMessageDialog(this, "Inserted Succesfully");
    }catch(ClassNotFoundException | SQLException e){OptionPane.showMessageDialog(this, e.getMessage());}
}

private void CancelActionPerformed(java.awt.event.ActionEvent evt) {
    dispose();
    Main main = new Main();
    main.show();
}
```

[drv.java](#)

Κώδικας:

```
public void updateTable(){
    try{
        Class.forName("com.mysql.cj.jdbc.Driver");
        Connection con = DriverManager.getConnection("jdbc:mysql://localhost:3306/travel_agency?zeroDateTimeBehavior=CONVERT_TO_NULL", "root", "root");
        String select="SELECT * FROM driver;";
        Statement slct = con.createStatement();
        ResultSet rs = slct.executeQuery(select);
        DefaultTableModel tbModel= (DefaultTableModel) DriverTable.getModel();
        tbModel.setNumRows(0);
        while(rs.next()){
            String at = rs.getString("drv_AT");
            String license = rs.getString("drv_license");
            String route = rs.getString("drv_route");
            String experience = rs.getString("drv_experience");
            String tb_data[]= {at,license,route,experience};
            tbModel.addRow(tb_data);
        }
    }
}
```

```

    }
    con.close();
} catch(ClassNotFoundException | SQLException e){JOptionPane.showMessageDialog(this, e.getMessage());}
}

public void updateCombo(){
    try{
        Class.forName("com.mysql.cj.jdbc.Driver");
        Connection con = DriverManager.getConnection("jdbc:mysql://localhost:3306/travel_agency?zeroDateTimeBehavior=CONVERT_TO_NULL", "root", "root");
        String select="SELECT wrk_AT FROM worker;";
        Statement slct = con.createStatement();
        ResultSet rs = slct.executeQuery(select);
        DefaultComboBoxModel mod = new DefaultComboBoxModel();
        mod.removeAllElements();
        while(rs.next()){mod.addElement(rs.getString("wrk_AT"));}
        wrk_AT.setModel(mod);
        con.close();
    } catch(ClassNotFoundException | SQLException e){JOptionPane.showMessageDialog(this, e.getMessage());}
}

private void formWindowOpened(java.awt.event.WindowEvent evt) {
    updateCombo();
    updateTable();
}

private void DriverTableMouseClicked(java.awt.event.MouseEvent evt) {
    DefaultTableModel tbModel= (DefaultTableModel) DriverTable.getModel();
    String at = tbModel.getValueAt(DriverTable.getSelectedRow(), 0).toString();
    String license = tbModel.getValueAt(DriverTable.getSelectedRow(), 1).toString();
    String route = tbModel.getValueAt(DriverTable.getSelectedRow(), 2).toString();
    String exp = tbModel.getValueAt(DriverTable.getSelectedRow(), 3).toString();
    wrk_AT.setSelectedItem(at);
    drv_license.setSelectedItem(license);
    drv_route.setSelectedItem(route);
    drv_experience.setText(exp);
}

private void InsertActionPerformed(java.awt.event.ActionEvent evt) {
    try{
        Class.forName("com.mysql.cj.jdbc.Driver");
        Connection con = DriverManager.getConnection("jdbc:mysql://localhost:3306/travel_agency?zeroDateTimeBehavior=CONVERT_TO_NULL", "root", "root");
        String insert="INSERT INTO driver(drv_AT,drv_license,drv_route,drv_experience) VALUES(?,?,?,?)";
        PreparedStatement insrt = con.prepareStatement(insert);
        insrt.setString(1,wrk_AT.getSelectedItem().toString());
        insrt.setString(2,drv_license.getSelectedItem().toString());
        insrt.setString(3,drv_route.getSelectedItem().toString());
        insrt.setInt(4,Integer.parseInt(drv_experience.getText()));
        insrt.execute();
        updateTable();
        JOptionPane.showMessageDialog(this, "Inserted Succesfully");
    } catch(ClassNotFoundException | SQLException e){JOptionPane.showMessageDialog(this, e.getMessage());}
}

private void CancelActionPerformed(java.awt.event.ActionEvent evt) {
    dispose();
    Main main = new Main();
    main.show();
}
}

```

[dst.java](#)

## Κώδικας:

```

public void updateTable(){
    try{
        Class.forName("com.mysql.cj.jdbc.Driver");
        Connection con = DriverManager.getConnection("jdbc:mysql://localhost:3306/travel_agency?zeroDateTimeBehavior=CONVERT_TO_NULL", "root", "root");
        String select="SELECT * FROM destination;";
        Statement slct = con.createStatement();
        ResultSet rs = slct.executeQuery(select);
        DefaultTableModel tbModel= (DefaultTableModel) DstTable.getModel();
        tbModel.setNumRows(0);
        while(rs.next()){
            String dst_id = rs.getString("dst_id");
            String name = rs.getString("dst_name");
            String description = rs.getString("dst_descr");
            String type = rs.getString("dst_rtype");
            String language = rs.getString("dst_language");
            String location = rs.getString("dst_location");
            String tb_data[] = {dst_id, name, description, type, language, location};

```

```

        tbModel.addRow(tb_data);
    }
    con.close();
}catch(ClassNotFoundException | SQLException e){JOptionPane.showMessageDialog(this, e.getMessage());}
}

private void formWindowOpened(java.awt.event.WindowEvent evt) {
    updateTable();
}

private void Update1ActionPerformed(java.awt.event.ActionEvent evt) {
    try{
        DefaultTableModel tbModel= (DefaultTableModel) DstTable.getModel();
        int at = Integer.parseInt(tbModel.getValueAt(DstTable.getSelectedRow(), 0).toString());
        Class.forName("com.mysql.cj.jdbc.Driver");
        Connection con = DriverManager.getConnection("jdbc:mysql://localhost:3306/travel_agency?zeroDateTimeBehavior=CONVERT_TO_NULL", "root", "root");
        String update="UPDATE destination SET dst_name = ?, dst_descr = ?, dst_rtype = ?, dst_language = ?, dst_location = ? WHERE dst_id = ?";
        PreparedStatement upd = con.prepareStatement(update);
        upd.setString(1, dst_name.getText());
        upd.setString(2, dst_descr.getText());
        upd.setString(3, dst_rtype.getSelectedItem().toString());
        upd.setString(4, dst_language.getText());
        upd.setInt(5, Integer.parseInt(dst_location.getText()));
        upd.setInt(6, at);
        upd.executeUpdate();
        updateTable();
        JOptionPane.showMessageDialog(this, "Updated Succesfully");
        con.close();
    }catch(ClassNotFoundException | SQLException e){JOptionPane.showMessageDialog(this, e.getMessage());}
}

private void Delete1ActionPerformed(java.awt.event.ActionEvent evt) {
    try{
        DefaultTableModel tbModel= (DefaultTableModel) DstTable.getModel();
        int at = Integer.parseInt(tbModel.getValueAt(DstTable.getSelectedRow(), 0).toString());
        Class.forName("com.mysql.cj.jdbc.Driver");
        Connection con = DriverManager.getConnection("jdbc:mysql://localhost:3306/travel_agency?zeroDateTimeBehavior=CONVERT_TO_NULL", "root", "root");
        String delete="DELETE FROM destination WHERE dst_id = ?";
        PreparedStatement dlt = con.prepareStatement(delete);
        dlt.setInt(1, at);
        dlt.executeUpdate();
        updateTable();
        dst_name.setText("");
        dst_descr.setText("");
        dst_language.setText("");
        dst_location.setText("");
        JOptionPane.showMessageDialog(this, "Deleted Succesfully");
        con.close();
    }catch(ClassNotFoundException | SQLException e){JOptionPane.showMessageDialog(this, e.getMessage());}
}

private void CancelActionPerformed(java.awt.event.ActionEvent evt) {
    dispose();
    Main main = new Main();
    main.show();
}

private void InsertActionPerformed(java.awt.event.ActionEvent evt) {
    try{
        Class.forName("com.mysql.cj.jdbc.Driver");
        Connection con = DriverManager.getConnection("jdbc:mysql://localhost:3306/travel_agency?zeroDateTimeBehavior=CONVERT_TO_NULL", "root", "root");
        String insert="INSERT INTO destination(dst_id,dst_name,dst_descr,dst_rtype,dst_language,dst_location) VALUES(null,?,?,?,?,?)";
        PreparedStatement insrt = con.prepareStatement(insert);
        insrt.setString(1,dst_name.getText());
        insrt.setString(2,dst_descr.getText());
        insrt.setString(3,dst_rtype.getSelectedItem().toString());
        insrt.setString(4,dst_language.getText());
        insrt.setInt(5,Integer.parseInt(dst_location.getText()));
        insrt.execute();
        updateTable();
        JOptionPane.showMessageDialog(this, "Inserted Succesfully");
    }catch(ClassNotFoundException | SQLException e){JOptionPane.showMessageDialog(this, e.getMessage());}
}

private void DstTableMouseClicked(java.awt.event.MouseEvent evt) {
    DefaultTableModel tbModel= (DefaultTableModel) DstTable.getModel();
    String name = tbModel.getValueAt(DstTable.getSelectedRow(), 1).toString();

```

```
String description = tbModel.getValueAt(DstTable.getSelectedRow(), 2).toString();
String type = tbModel.getValueAt(DstTable.getSelectedRow(), 3).toString();
String language = tbModel.getValueAt(DstTable.getSelectedRow(), 4).toString();
String location = tbModel.getValueAt(DstTable.getSelectedRow(), 5).toString();
dst_name.setText(name);
dst_descr.setText(description);
dst_rtype.setSelectedItem(type);
dst_language.setText(language);
dst_location.setText(location);
}
```

[evt.java](#)

## Κώδικας:

```
public void convertTimeStamp(){
    String time1 = PickTime1.getText();
    String time2 = PickTime2.getText();
    time1=time1.replace(" ", "");
    time2=time2.replace(" ", "");
    if(time1.contains("PM")){
        String[] parts = time1.split(":");
        String part1 = parts[0];
        String part2 = parts[1];
        if(!"12".equals(part1)){
            int part1int = Integer.parseInt(part1)+12;
            part1 = Integer.toString(part1int);
            time1 = part1.concat(":").concat(part2);
        }else time1 = part1.concat(":").concat(part2);
    }else if(time1.contains("AM")){
        String[] parts = time1.split(":");
        String part1 = parts[0];
        String part2 = parts[1];
        if("12".equals(part1)){
            part1 = "00";
            time1 = part1.concat(":").concat(part2);
        }else time1 = part1.concat(":").concat(part2);
    }else{
        String[] parts = time1.split(":");
        String part1 = parts[0];
        String part2 = parts[1];
        time1 = part1.concat(":").concat(part2);
    }
    time1=time1.replace("PM", "");
    time1=time1.replace("AM", "");

    if(time2.contains("PM")){
        String[] parts = time2.split(":");
        String part1 = parts[0];
        String part2 = parts[1];
        if(!"12".equals(part1)){
            int part1int = Integer.parseInt(part1)+12;
            part1 = Integer.toString(part1int);
            time2 = part1.concat(":").concat(part2);
        }else time2 = part1.concat(":").concat(part2);
    }else if(time2.contains("AM")){
        String[] parts = time2.split(":");
        String part1 = parts[0];
        String part2 = parts[1];
        if("12".equals(part1)){
            part1 = "00";
            time2 = part1.concat(":").concat(part2);
        }else time2 = part1.concat(":").concat(part2);
    }
    time2=time2.replace("PM", "");
    time2=time2.replace("AM", "");
    time1=time1.concat(":00");
    time2=time2.concat(":00");
    DateFormat dateFormat1 = new SimpleDateFormat("yyyy-MM-dd ");
    String strDate1 = dateFormat1.format(jDateChooser1.getDate());
    String date1 = strDate1.concat(time1);
    DateFormat dateFormat2 = new SimpleDateFormat("yyyy-MM-dd ");
    String strDate2 = dateFormat2.format(jDateChooser2.getDate());
    String date2 = strDate2.concat(time2);
    System.out.println(date1);
    t1 = java.sql.Timestamp.valueOf(date1);
    t2 = java.sql.Timestamp.valueOf(date2);
}
```

```

}

public void updateTable(){
    try{
        Class.forName("com.mysql.cj.jdbc.Driver");
        Connection con = DriverManager.getConnection("jdbc:mysql://localhost:3306/travel_agency?zeroDateTimeBehavior=CONVERT_TO_NULL", "root", "root");
        String select="SELECT * FROM event;";
        Statement slct = con.createStatement();
        ResultSet rs = slct.executeQuery(select);
        DefaultTableModel tbModel= (DefaultTableModel) EventTable.getModel();
        tbModel.setNumRows(0);
        while(rs.next()){
            String id = rs.getString("ev_tr_id");
            String start = rs.getString("ev_start");
            String end = rs.getString("ev_end");
            String descr = rs.getString("ev_descr");
            String tb_data[] = {id, start, end, descr};
            tbModel.addRow(tb_data);
        }
        con.close();
    }catch(ClassNotFoundException | SQLException e){JOptionPane.showMessageDialog(this, e.getMessage());}
}

public void updateCombo(){
    PickTime1.setText("Set Time");
    PickTime2.setText("Set Time");
    try{
        Class.forName("com.mysql.cj.jdbc.Driver");
        Connection con = DriverManager.getConnection("jdbc:mysql://localhost:3306/travel_agency?zeroDateTimeBehavior=CONVERT_TO_NULL", "root", "root");
        String select="SELECT DISTINCT ev_tr_id FROM event;";
        Statement slct = con.createStatement();
        ResultSet rs = slct.executeQuery(select);
        DefaultComboBoxModel mod = new DefaultComboBoxModel();
        mod.removeAllElements();
        while(rs.next()){
            String box = "Trip: "+rs.getInt("ev_tr_id")+"";
            mod.addElement(box);
        }
        ev_tr_id.setModel(mod);
        con.close();
    }catch(ClassNotFoundException | SQLException e){JOptionPane.showMessageDialog(this, e.getMessage());}
}

private void PickTime1MouseClicked(java.awt.event.MouseEvent evt) {
    timePicker1.showPopup(this, 100, 100);
    String time = timePicker1.getSelectedTime();
    time=time.replace("MP", "PM");
    PickTime1.setText(time);
}

private void formWindowOpened(java.awt.event.WindowEvent evt) {
    updateCombo();
    updateTable();
}

private void PickTime2MouseClicked(java.awt.event.MouseEvent evt) {
    timePicker2.showPopup(this, 100, 100);
    String time = timePicker2.getSelectedTime();
    time=time.replace("MP", "PM");
    PickTime2.setText(time);
}

private void EventTableMouseClicked(java.awt.event.MouseEvent evt) {
    DefaultTableModel tbModel= (DefaultTableModel) EventTable.getModel();
    String id = "Trip: "+tbModel.getValueAt(EventTable.getSelectedRow(), 0).toString()+"";
    String start = tbModel.getValueAt(EventTable.getSelectedRow(), 1).toString();
    String end = tbModel.getValueAt(EventTable.getSelectedRow(), 2).toString();
    String descr = tbModel.getValueAt(EventTable.getSelectedRow(), 3).toString();

    temp = java.sql.Timestamp.valueOf(start);

    ev_tr_id.setSelectedItem(id);
    ev_descr.setText(descr);
    String[] parts1 = start.split(" ");
    String part1s = parts1[0];
    String part2s = parts1[1];

    jDateChooser1.setDate(java.sql.Date.valueOf(part1s));

```

```

PickTime1.setText(part2s);

String[] parts2 = end.split(" ");
String part1e = parts2[0];
String part2e = parts2[1];
jDateChooser2.setDate(java.sql.Date.valueOf(part1e));
PickTime2.setText(part2e);
}

private void InsertActionPerformed(java.awt.event.ActionEvent evt) {
    try{
        Class.forName("com.mysql.cj.jdbc.Driver");
        Connection con = DriverManager.getConnection("jdbc:mysql://localhost:3306/travel_agency?zeroDateTimeBehavior=CONVERT_TO_NULL", "root", "root");
        String insert="INSERT INTO event(ev_tr_id,ev_start,ev_end,ev_descr) VALUES(?,?,?,?)";
        PreparedStatement insrt = con.prepareStatement(insert);
        convertTimeStamp();
        String code = ev_tr_id.getSelectedItem().toString().replaceAll("[^0-9]", "");
        int i=Integer.parseInt(code);
        insrt.setInt(1, i);
        insrt.setTimestamp(2,t1);
        insrt.setTimestamp(3,t2);
        insrt.setString(4,ev_descr.getText());
        insrt.execute();
        updateTable();
        JOptionPane.showMessageDialog(this, "Inserted Succesfully");
    }catch(ClassNotFoundException | SQLException e){JOptionPane.showMessageDialog(this, e.getMessage());}
}

private void CancelActionPerformed(java.awt.event.ActionEvent evt) {
    dispose();
    Main main = new Main();
    main.show();
}

private void DeleteActionPerformed(java.awt.event.ActionEvent evt) {
    try{
        Class.forName("com.mysql.cj.jdbc.Driver");
        Connection con = DriverManager.getConnection("jdbc:mysql://localhost:3306/travel_agency?zeroDateTimeBehavior=CONVERT_TO_NULL", "root", "root");
        String delete="DELETE FROM event WHERE ev_tr_id = ?";
        PreparedStatement dlt = con.prepareStatement(delete);
        String code = ev_tr_id.getSelectedItem().toString().replaceAll("[^0-9]", "");
        int i=Integer.parseInt(code);
        dlt.setInt(1, i);
        dlt.executeUpdate();
        updateTable();
        jDateChooser1.setDate(null);
        jDateChooser2.setDate(null);
        PickTime1.setText("");
        PickTime2.setText("");
        ev_descr.setText("");
        JOptionPane.showMessageDialog(this, "Deleted Succesfully");
        con.close();
    }catch(ClassNotFoundException | SQLException e){JOptionPane.showMessageDialog(this, e.getMessage());}
}

private void UpdateActionPerformed(java.awt.event.ActionEvent evt) {
    convertTimeStamp();
    System.out.println(temp);
    System.out.println(t1);
    if(temp.toString().equals(t1.toString())){
        try{
            Class.forName("com.mysql.cj.jdbc.Driver");
            Connection con = DriverManager.getConnection("jdbc:mysql://localhost:3306/travel_agency?zeroDateTimeBehavior=CONVERT_TO_NULL", "root", "root");
            String update="UPDATE event SET ev_end = ?, ev_descr = ? WHERE ev_tr_id = ? AND ev_start = ?";
            PreparedStatement upd = con.prepareStatement(update);
            String code = ev_tr_id.getSelectedItem().toString().replaceAll("[^0-9]", "");
            int i=Integer.parseInt(code);
            upd.setTimestamp(1, t2);
            upd.setString(2, ev_descr.getText());
            upd.setInt(3, i);
            upd.setTimestamp(4, t1);
            upd.executeUpdate();
            updateTable();
            JOptionPane.showMessageDialog(this, "Updated Succesfully");
            con.close();
        }catch(ClassNotFoundException | SQLException e){JOptionPane.showMessageDialog(this, e.getMessage());}
    }else{JOptionPane.showMessageDialog(this, "You cannot Update the Event Start Date! Try Using Insert/Delete Instead.");}
}

```

```
}
```

[gui.java](#)

**Κώδικας:**

```
public void updateTable(){
    try{
        Class.forName("com.mysql.cj.jdbc.Driver");
        Connection con = DriverManager.getConnection("jdbc:mysql://localhost:3306/travel_agency?zeroDateTimeBehavior=CONVERT_TO_NULL", "root", "root");
        String select2="SELECT * FROM guide;";
        Statement slct2 = con.createStatement();
        ResultSet rs2 = slct2.executeQuery(select2);
        DefaultTableModel tbModel= (DefaultTableModel) GuiTable.getModel();
        tbModel.setNumRows(0);
        while(rs2.next()){
            String at = rs2.getString("gui_AT");
            String cv = rs2.getString("gui_cv");
            String tb_data[] = {at,cv};
            tbModel.addRow(tb_data);
        }
        con.close();
    }catch(ClassNotFoundException | SQLException e){JOptionPane.showMessageDialog(this, e.getMessage());}
}

public void updateCombo(){
    try{
        Class.forName("com.mysql.cj.jdbc.Driver");
        Connection con = DriverManager.getConnection("jdbc:mysql://localhost:3306/travel_agency?zeroDateTimeBehavior=CONVERT_TO_NULL", "root", "root");
        String select="SELECT wrk_AT FROM worker;";
        Statement slct = con.createStatement();
        ResultSet rs = slct.executeQuery(select);
        DefaultComboBoxModel mod = new DefaultComboBoxModel();
        mod.removeAllElements();
        while(rs.next()){mod.addElement(rs.getString("wrk_AT"));}
        updateTable();
        gui_AT.setModel(mod);
        con.close();
    }catch(ClassNotFoundException | SQLException e){JOptionPane.showMessageDialog(this, e.getMessage());}
}

private void formWindowOpened(java.awt.event.WindowEvent evt) {
    updateCombo();
}

private void GuiTableMouseClicked(java.awt.event.MouseEvent evt) {
    DefaultTableModel tbModel= (DefaultTableModel) GuiTable.getModel();
    String gui_at = tbModel.getValueAt(GuiTable.getSelectedRow(), 0).toString();
    String gui_CV = tbModel.getValueAt(GuiTable.getSelectedRow(), 1).toString();
    gui_AT.setSelectedItem(gui_at);
    gui_cv.setText(gui_CV);
}

private void InsertActionPerformed(java.awt.event.ActionEvent evt) {
    try{
        Class.forName("com.mysql.cj.jdbc.Driver");
        Connection con = DriverManager.getConnection("jdbc:mysql://localhost:3306/travel_agency?zeroDateTimeBehavior=CONVERT_TO_NULL", "root", "root");
        String insert="INSERT INTO guide(gui_AT,gui_cv) VALUES(?,?)";
        PreparedStatement insrt = con.prepareStatement(insert);
        insrt.setString(1,gui_AT.getSelectedItem().toString());
        insrt.setString(2,gui_cv.getText());
        insrt.execute();
        updateTable();
        JOptionPane.showMessageDialog(this, "Inserted Succesfully");
    }catch(ClassNotFoundException | SQLException e){JOptionPane.showMessageDialog(this, e.getMessage());}
}

private void CancelActionPerformed(java.awt.event.ActionEvent evt) {
    dispose();
    Main main = new Main();
    main.show();
}

private void Delete1ActionPerformed(java.awt.event.ActionEvent evt) {
    try{
        Class.forName("com.mysql.cj.jdbc.Driver");
        Connection con = DriverManager.getConnection("jdbc:mysql://localhost:3306/travel_agency?zeroDateTimeBehavior=CONVERT_TO_NULL", "root", "root");
        String delete="DELETE FROM guide WHERE gui_AT = ?";
        PreparedStatement dlt = con.prepareStatement(delete);
```

```

        dlt.setString(1, gui_AT.getSelectedItemAt().toString());
        dlt.executeUpdate();
        updateTable();
        gui_cv.setText("");
        JOptionPane.showMessageDialog(this, "Deleted Succesfully");
        con.close();
    }catch(ClassNotFoundException | SQLException e){JOptionPane.showMessageDialog(this, e.getMessage());}
}

private void Update1ActionPerformed(java.awt.event.ActionEvent evt) {
    try{
        Class.forName("com.mysql.cj.jdbc.Driver");
        Connection con = DriverManager.getConnection("jdbc:mysql://localhost:3306/travel_agency?zeroDateTimeBehavior=CONVERT_TO_NULL", "root", "root");
        String update="UPDATE guide SET gui_cv = ? WHERE gui_AT = ?";
        PreparedStatement upd = con.prepareStatement(update);
        upd.setString(1, gui_cv.getText());
        upd.setString(2, gui_AT.getSelectedItemAt().toString());
        upd.executeUpdate();
        updateTable();
        JOptionPane.showMessageDialog(this, "Updated Succesfully");
        con.close();
    }catch(ClassNotFoundException | SQLException e){JOptionPane.showMessageDialog(this, e.getMessage());}
}

```

[it.java](#)

## Κώδικας:

```

static Timestamp t1;
static Timestamp t2;
static String temp;
public void convertTimeStamp(){
    String time1 = PickTime1.getText();
    String time2 = PickTime2.getText();
    time1=time1.replace(" ", "");
    time2=time2.replace(" ", "");
    if(time1.contains("PM")){
        String[] parts = time1.split(":");
        String part1 = parts[0];
        String part2 = parts[1];
        if(!"12".equals(part1)){
            int part1int = Integer.parseInt(part1)+12;
            part1 = Integer.toString(part1int);
            time1 = part1.concat(":").concat(part2);
        }else time1 = part1.concat(":").concat(part2);
    }else if(time1.contains("AM")){
        String[] parts = time1.split(":");
        String part1 = parts[0];
        String part2 = parts[1];
        if("12".equals(part1)){
            part1 = "00";
            time1 = part1.concat(":").concat(part2);
        }else time1 = part1.concat(":").concat(part2);
    }else{
        String[] parts = time1.split(":");
        String part1 = parts[0];
        String part2 = parts[1];
        time1 = part1.concat(":").concat(part2);
    }
    time1=time1.replace("PM", "");
    time1=time1.replace("AM", "");
    if(isRetired.isSelected()){
        if(time2.contains("PM")){
            String[] parts = time2.split(":");
            String part1 = parts[0];
            String part2 = parts[1];
            if(!"12".equals(part1)){
                int part1int = Integer.parseInt(part1)+12;
                part1 = Integer.toString(part1int);
                time2 = part1.concat(":").concat(part2);
            }else time2 = part1.concat(":").concat(part2);
        }else{
            String[] parts = time2.split(":");
            String part1 = parts[0];
            String part2 = parts[1];
            if("12".equals(part1)){
                part1 = "00";

```



```

        time2 = part1.concat(":").concat(part2);
    }else time2 = part1.concat(":".concat(part2);
    }
    time2=time2.replace("PM", "");
    time2=time2.replace("AM", "");
    time2=time2.concat(":00");
    DateFormat dateFormat2 = new SimpleDateFormat("yyyy-MM-dd ");
    String strDate2 = dateFormat2.format(jDateChooser2.getDate());
    String date2 = strDate2.concat(time2);
    t2 = java.sql.Timestamp.valueOf(date2);
}else t2 = null;

time1=time1.concat(":00");
DateFormat dateFormat1 = new SimpleDateFormat("yyyy-MM-dd ");
String strDate1 = dateFormat1.format(jDateChooser1.getDate());
String date1 = strDate1.concat(time1);
t1 = java.sql.Timestamp.valueOf(date1);
}

public void updateTable(){
    try{
        Class.forName("com.mysql.cj.jdbc.Driver");
        Connection con = DriverManager.getConnection("jdbc:mysql://localhost:3306/travel_agency?zeroDateTimeBehavior=CONVERT_TO_NULL", "root", "root");
        String select="SELECT * FROM it;";
        Statement slct = con.createStatement();
        ResultSet rs = slct.executeQuery(select);
        DefaultTableModel tbModel= (DefaultTableModel) itTable.getModel();
        tbModel.setNumRows(0);
        while(rs.next()){
            String wrkat = rs.getString("wrk_it_AT");
            String at = rs.getString("IT_AT");
            String password = rs.getString("pass");
            String start = rs.getString("start_date");
            String end = rs.getString("end_date");
            String tb_data[]= {wrkat, at, password, start, end};
            tbModel.addRow(tb_data);
        }
        con.close();
    }catch(ClassNotFoundException | SQLException e){JOptionPane.showMessageDialog(this, e.getMessage());}
}

public void updateCombo(){
    jLabel5.setVisible(false);
    jDateChooser2.setVisible(false);
    PickTime2.setVisible(false);
    try
    {
        Class.forName("com.mysql.cj.jdbc.Driver");
        Connection con = DriverManager.getConnection("jdbc:mysql://localhost:3306/travel_agency?zeroDateTimeBehavior=CONVERT_TO_NULL", "root", "root");
        String select="SELECT wrk_AT FROM worker;";
        Statement slct = con.createStatement();
        ResultSet rs = slct.executeQuery(select);
        DefaultComboBoxModel mod = new DefaultComboBoxModel();
        mod.removeAllElements();
        while(rs.next()){
            mod.addElement(rs.getString("wrk_AT"));
        }
        wrk_it_AT.setModel(mod);
        con.close();
    }catch(ClassNotFoundException | SQLException e){
        JOptionPane.showMessageDialog(this, e.getMessage());
    }
    isRetired.addItemListener(new ItemListener() {
        @Override
        public void itemStateChanged(ItemEvent e) {
            if(e.getStateChange() == ItemEvent.SELECTED) {
                jLabel5.setVisible(true);
                jDateChooser2.setVisible(true);
                PickTime2.setVisible(true);
            }else{
                jLabel5.setVisible(false);
                jDateChooser2.setVisible(false);
                PickTime2.setVisible(false);
            }
        }
    });
}

private void PickTime2MouseClicked(java.awt.event.MouseEvent evt) {

```

```

timePicker2.showPopup(this, 100, 100);
String time = timePicker2.getSelectedTime();
time=time.replace("MP", "PM");
PickTime2.setText(time);
}

private void PickTime1MouseClicked(java.awt.event.MouseEvent evt) {
    timePicker1.showPopup(this, 100, 100);
    String time = timePicker1.getSelectedTime();
    time=time.replace("MP", "PM");
    PickTime1.setText(time);
}

private void formWindowOpened(java.awt.event.WindowEvent evt) {
    updateCombo();
    updateTable();
}

private void itTableMouseClicked(java.awt.event.MouseEvent evt) {
    DefaultTableModel tbModel= (DefaultTableModel) itTable.getModel();
    String wrkat = tbModel.getValueAt(itTable.getSelectedRow(), 0).toString();
    String itat = tbModel.getValueAt(itTable.getSelectedRow(), 1).toString();
    String password = tbModel.getValueAt(itTable.getSelectedRow(), 2).toString();
    String start = tbModel.getValueAt(itTable.getSelectedRow(), 3).toString();

    if(tbModel.getValueAt(itTable.getSelectedRow(), 4) != null){
        String end = tbModel.getValueAt(itTable.getSelectedRow(), 4).toString();
        String[] parts2 = end.split(" ");
        String part1e = parts2[0];
        String part2e = parts2[1];
        jDateChooser2.setDate(java.sql.Date.valueOf(part1e));
        PickTime2.setText(part2e);
    }else{
        t2 = null;
        jDateChooser2.setDate(null);
        PickTime2.setText("");
        isRetired.setSelected(false);
    }
    if(tbModel.getValueAt(itTable.getSelectedRow(), 4)!= null){
        isRetired.setSelected(true);
    }
    wrk_it_AT.setSelectedItem(wrkat);
    IT_AT.setText(itat);
    pass.setText(password);
    temp = tbModel.getValueAt(itTable.getSelectedRow(), 1).toString();
    String[] parts1 = start.split(" ");
    String part1s = parts1[0];
    String part2s = parts1[1];

    jDateChooser1.setDate(java.sql.Date.valueOf(part1s));
    PickTime1.setText(part2s);
}

private void InsertActionPerformed(java.awt.event.ActionEvent evt) {
    try{
        Class.forName("com.mysql.cj.jdbc.Driver");
        Connection con = DriverManager.getConnection("jdbc:mysql://localhost:3306/travel_agency?zeroDateTimeBehavior=CONVERT_TO_NULL", "root", "root");
        String insert="INSERT INTO it(wrk_it_AT,IT_AT,pass,start_date,end_date) VALUES(?,?,?,?,?)";
        PreparedStatement insrt = con.prepareStatement(insert);
        convertTimeStamp();
        insrt.setString(1, wrk_it_AT.getSelectedItem().toString());
        insrt.setString(2, IT_AT.getText());
        insrt.setString(3, pass.getText());
        insrt.setTimestamp(4,t1);
        if(isRetired.isSelected()){insrt.setTimestamp(5,t2);}
        else insrt.setTimestamp(5, t2);
        insrt.execute();
        updateTable();
        JOptionPane.showMessageDialog(this, "Inserted Succesfully");
    }catch(ClassNotFoundException | SQLException e){JOptionPane.showMessageDialog(this, e.getMessage());}
}

private void CancelActionPerformed(java.awt.event.ActionEvent evt) {
    dispose();
    Main main = new Main();
    main.show();
}

```

```

private void Delete1ActionPerformed(java.awt.event.ActionEvent evt) {
    try{
        Class.forName("com.mysql.cj.jdbc.Driver");
        Connection con = DriverManager.getConnection("jdbc:mysql://localhost:3306/travel_agency?zeroDateTimeBehavior=CONVERT_TO_NULL", "root", "root");
        String delete="DELETE FROM it WHERE IT_AT = ?";
        PreparedStatement dlt = con.prepareStatement(delete);
        dlt.setString(1, IT_AT.getText());
        dlt.executeUpdate();
        updateTable();
        t1 = null;
        t2 = null;
        pass.setText("");
        jDateChooser1.setDate(null);
        jDateChooser2.setDate(null);
        PickTime1.setText("");
        PickTime2.setText("");
        isRetired.setSelected(false);
        IT_AT.setText("");
        JOptionPane.showMessageDialog(this, "Deleted Succesfully");
        con.close();
    }catch(ClassNotFoundException | SQLException e){JOptionPane.showMessageDialog(this, e.getMessage());}
}

private void Update1ActionPerformed(java.awt.event.ActionEvent evt) {
    convertTimeStamp();
    if(temp.equals(IT_AT.getText())){
        try{
            Class.forName("com.mysql.cj.jdbc.Driver");
            Connection con = DriverManager.getConnection("jdbc:mysql://localhost:3306/travel_agency?zeroDateTimeBehavior=CONVERT_TO_NULL", "root", "root");
            String update="UPDATE it SET pass = ?, start_date = ?, end_date = ? WHERE wrk_it_AT = ? AND IT_AT = ?";
            PreparedStatement upd = con.prepareStatement(update);
            upd.setString(1, pass.getText());
            upd.setTimestamp(2, t1);
            upd.setTimestamp(3, t2);
            upd.setString(4, wrk_it_AT.getSelectedItem().toString());
            upd.setString(5, IT_AT.getText());
            upd.executeUpdate();
            updateTable();
            JOptionPane.showMessageDialog(this, "Updated Succesfully");
            con.close();
        }catch(ClassNotFoundException | SQLException e){JOptionPane.showMessageDialog(this, e.getMessage());}
    }else{JOptionPane.showMessageDialog(this, "You cannot Update the Travel_to destination ID! Try Using Insert/Delete Instead.");}
}

```

[lang.java](#)

### Κώδικας:

```

public void updateTable(){
    try{
        Class.forName("com.mysql.cj.jdbc.Driver");
        Connection con = DriverManager.getConnection("jdbc:mysql://localhost:3306/travel_agency?zeroDateTimeBehavior=CONVERT_TO_NULL", "root", "root");
        String select="SELECT * FROM languages;";
        Statement slct = con.createStatement();
        ResultSet rs = slct.executeQuery(select);
        DefaultTableModel tbModel= (DefaultTableModel) LanguageTable.getModel();
        tbModel.setNumRows(0);
        while(rs.next()){
            String at = rs.getString("Ing_gui_AT");
            String language = rs.getString("Ing_language");
            String tb_data[] = {at,language};
            tbModel.addRow(tb_data);
        }
        con.close();
    }catch(ClassNotFoundException | SQLException e){JOptionPane.showMessageDialog(this, e.getMessage());}
}

public void updateCombo(){
    try{
        Class.forName("com.mysql.cj.jdbc.Driver");
        Connection con = DriverManager.getConnection("jdbc:mysql://localhost:3306/travel_agency?zeroDateTimeBehavior=CONVERT_TO_NULL", "root", "root");
        String select="SELECT wrk_AT FROM worker INNER JOIN guide ON wrk_AT = gui_AT;";
        Statement slct = con.createStatement();
        ResultSet rs = slct.executeQuery(select);
        DefaultComboBoxModel mod = new DefaultComboBoxModel();
        mod.removeAllElements();
        while(rs.next()){mod.addElement(rs.getString("wrk_AT"));}
        Ing_gui_AT.setModel(mod);
        con.close();
    }
}

```

```

    }catch(ClassNotFoundException | SQLException e){OptionPane.showMessageDialog(this, e.getMessage());}
}

private void formWindowOpened(java.awt.event.WindowEvent evt) {
    updateCombo();
    updateTable();
}

private void LanguageTableMouseClicked(java.awt.event.MouseEvent evt) {
    DefaultTableModel tbModel= (DefaultTableModel) LanguageTable.getModel();
    String code = tbModel.getValueAt(LanguageTable.getSelectedRow(), 0).toString();
    String language = tbModel.getValueAt(LanguageTable.getSelectedRow(), 1).toString();
    lng_gui_AT.setSelectedItem(code);
    lng_language.setText(language);
}

private void InsertActionPerformed(java.awt.event.ActionEvent evt) {
    try{
        Class.forName("com.mysql.cj.jdbc.Driver");
        Connection con = DriverManager.getConnection("jdbc:mysql://localhost:3306/travel_agency?zeroDateTimeBehavior=CONVERT_TO_NULL", "root", "root");
        String insert="INSERT INTO languages(lng_gui_AT,lng_language) VALUES(?,?)";
        PreparedStatement insrt = con.prepareStatement(insert);
        insrt.setString(1,lng_gui_AT.getSelectedItem().toString());
        insrt.setString(2,lng_language.getText());
        insrt.execute();
        updateTable();
        JOptionPane.showMessageDialog(this, "Inserted Succesfully");
    }catch(ClassNotFoundException | SQLException e){OptionPane.showMessageDialog(this, e.getMessage());}
}

private void CancelActionPerformed(java.awt.event.ActionEvent evt) {
    dispose();
    Main main = new Main();
    main.show();
}

private void Delete1ActionPerformed(java.awt.event.ActionEvent evt) {
    try{
        Class.forName("com.mysql.cj.jdbc.Driver");
        Connection con = DriverManager.getConnection("jdbc:mysql://localhost:3306/travel_agency?zeroDateTimeBehavior=CONVERT_TO_NULL", "root", "root");
        String delete="DELETE FROM languages WHERE lng_gui_AT = ? AND lng_language = ?";
        PreparedStatement dlt = con.prepareStatement(delete);
        dlt.setString(1,lng_gui_AT.getSelectedItem().toString());
        dlt.setString(2,lng_language.getText());
        dlt.execute();
        updateTable();
        lng_language.setText("");
        JOptionPane.showMessageDialog(this, "Deleted Succesfully");
    }catch(ClassNotFoundException | SQLException e){OptionPane.showMessageDialog(this, e.getMessage());}
}

private void Update1ActionPerformed(java.awt.event.ActionEvent evt) {
    JOptionPane.showMessageDialog(this, "You cannot Update this Table! Try Using Insert/Delete Instead.");
}

```

## mngs.java

### Κώδικας:

```

public void updateTable(){
    try{
        Class.forName("com.mysql.cj.jdbc.Driver");
        Connection con = DriverManager.getConnection("jdbc:mysql://localhost:3306/travel_agency?zeroDateTimeBehavior=CONVERT_TO_NULL", "root", "root");
        String select="SELECT * FROM manages;";
        Statement slct = con.createStatement();
        ResultSet rs = slct.executeQuery(select);
        DefaultTableModel tbModel= (DefaultTableModel) ManagesTable.getModel();
        tbModel.setNumRows(0);
        while(rs.next()){
            String at = rs.getString("mng_adm_AT");
            String code = rs.getString("mng_br_code");
            String pass = rs.getString("mng_pass");
            String user = rs.getString("mng_username");
            String tb_data[] = {at,code,pass,user};
            tbModel.addRow(tb_data);
        }
        con.close();
    }catch(ClassNotFoundException | SQLException e){OptionPane.showMessageDialog(this, e.getMessage());}
}

```

```

}
public void updateCombo(){
    try{
        Class.forName("com.mysql.cj.jdbc.Driver");
        Connection con = DriverManager.getConnection("jdbc:mysql://localhost:3306/travel_agency?zeroDateTimeBehavior=CONVERT_TO_NULL", "root", "root");
        String select1="SELECT br_code FROM branch;";
        String select2="SELECT adm_AT FROM admin;";
        Statement slct1 = con.createStatement();
        Statement slct2 = con.createStatement();
        ResultSet rs1 = slct1.executeQuery(select1);
        ResultSet rs2 = slct2.executeQuery(select2);
        DefaultComboBoxModel mod1 = new DefaultComboBoxModel();
        DefaultComboBoxModel mod2 = new DefaultComboBoxModel();
        mod1.removeAllElements();
        mod2.removeAllElements();
        while(rs1.next()){
            String box = "Branch: "+rs1.getInt("br_code")+"";
            mod1.addElement(box);
        }
        while(rs2.next()){
            mod2.addElement(rs2.getString("adm_AT"));
        }
        br_code.setModel(mod1);
        adm_AT.setModel(mod2);
        con.close();
    }catch(ClassNotFoundException | SQLException e){OptionPane.showMessageDialog(this, e.getMessage());}
}

private void formWindowOpened(java.awt.event.WindowEvent evt) {
    updateTable();
    updateCombo();
}

private void ManagesTableMouseClicked(java.awt.event.MouseEvent evt) {
    DefaultTableModel tbModel= (DefaultTableModel) ManagesTable.getModel();
    String at = tbModel.getValueAt(ManagesTable.getSelectedRow(), 0).toString();
    String code = tbModel.getValueAt(ManagesTable.getSelectedRow(), 1).toString();
    String pass = tbModel.getValueAt(ManagesTable.getSelectedRow(), 2).toString();
    String user = tbModel.getValueAt(ManagesTable.getSelectedRow(), 3).toString();
    adm_AT.setSelectedItem(at);
    br_code.setSelectedItem("Branch: "+code+"");
    mng_pass.setText(pass);
    mng_username.setText(user);
}

private void InsertActionPerformed(java.awt.event.ActionEvent evt) {
    try{
        Class.forName("com.mysql.cj.jdbc.Driver");
        Connection con = DriverManager.getConnection("jdbc:mysql://localhost:3306/travel_agency?zeroDateTimeBehavior=CONVERT_TO_NULL", "root", "root");
        String insert="INSERT INTO manages(mng_adm_AT,mng_br_code,mng_pass,mng_username) VALUES(?,?,?,?)";
        PreparedStatement insrt = con.prepareStatement(insert);
        String code = br_code.getSelectedItem().toString().replaceAll("[^0-9]", "");
        int i=Integer.parseInt(code);
        insrt.setString(1,adm_AT.getSelectedItem().toString());
        insrt.setInt(2,i);
        insrt.setString(3,mng_pass.getText());
        insrt.setString(4,mng_username.getText());
        insrt.execute();
        updateTable();
        JOptionPane.showMessageDialog(this, "Inserted Succesfully");
    }catch(ClassNotFoundException | SQLException e){OptionPane.showMessageDialog(this, e.getMessage());}
}

private void CancelActionPerformed(java.awt.event.ActionEvent evt) {
    dispose();
    Main main = new Main();
    main.show();
}

private void DeleteActionPerformed(java.awt.event.ActionEvent evt) {
    try{
        Class.forName("com.mysql.cj.jdbc.Driver");
        Connection con = DriverManager.getConnection("jdbc:mysql://localhost:3306/travel_agency?zeroDateTimeBehavior=CONVERT_TO_NULL", "root", "root");
        String delete="DELETE FROM manages WHERE mng_adm_AT = ?";
        PreparedStatement dlt = con.prepareStatement(delete);
        dlt.setString(1,adm_AT.getSelectedItem().toString());
        dlt.execute();
        updateTable();
    }
}

```

```

JOptionPane.showMessageDialog(this, "Deleted Successfully");
}catch(ClassNotFoundException | SQLException e){JOptionPane.showMessageDialog(this, e.getMessage());}
}

private void UpdateActionPerformed(java.awt.event.ActionEvent evt) {
try{
    Class.forName("com.mysql.cj.jdbc.Driver");
    Connection con = DriverManager.getConnection("jdbc:mysql://localhost:3306/travel_agency?zeroDateTimeBehavior=CONVERT_TO_NULL", "root", "root");
    String update="UPDATE manages SET mng_pass = ?, mng_username = ? WHERE mng_adm_AT = ? AND mng_br_code = ?";
    PreparedStatement upd = con.prepareStatement(update);
    int i = Integer.parseInt(br_code.getSelectedItem().toString().replaceAll("[^0-9]", ""));
    upd.setString(1,mng_pass.getText());
    upd.setString(2, mng_username.getText());
    upd.setString(3, adm_AT.getSelectedItem().toString());
    upd.setInt(4,i);
    upd.executeUpdate();
    updateTable();
    JOptionPane.showMessageDialog(this, "Updated Successfully");
    con.close();
}catch(ClassNotFoundException | SQLException e){JOptionPane.showMessageDialog(this, e.getMessage());}
}

```

[off.java](#)

## Κώδικας:

```

static Timestamp t1;
static Timestamp t2;
public void convertTimeStamp(){
    String time1 = PickTime1.getText();
    String time2 = PickTime2.getText();
    time1=time1.replace(" ", "");
    time2=time2.replace(" ", "");
    if(time1.contains("PM")){
        String[] parts = time1.split(":");
        String part1 = parts[0];
        String part2 = parts[1];
        if(!"12".equals(part1)){
            int part1int = Integer.parseInt(part1)+12;
            part1 = Integer.toString(part1int);
            time1 = part1.concat(":").concat(part2);
        }else time1 = part1.concat(":").concat(part2);
    }else if(time1.contains("AM")){
        String[] parts = time1.split(":");
        String part1 = parts[0];
        String part2 = parts[1];
        if("12".equals(part1)){
            part1 = "00";
            time1 = part1.concat(":").concat(part2);
        }else time1 = part1.concat(":").concat(part2);
    }else{
        String[] parts = time1.split(":");
        String part1 = parts[0];
        String part2 = parts[1];
        time1 = part1.concat(":").concat(part2);
    }
    time1=time1.replace("PM", "");
    time1=time1.replace("AM", "");

    if(time2.contains("PM")){
        String[] parts = time2.split(":");
        String part1 = parts[0];
        String part2 = parts[1];
        if(!"12".equals(part1)){
            int part1int = Integer.parseInt(part1)+12;
            part1 = Integer.toString(part1int);
            time2 = part1.concat(":").concat(part2);
        }else time2 = part1.concat(":").concat(part2);
    }else{
        String[] parts = time2.split(":");
        String part1 = parts[0];
        String part2 = parts[1];
        if("12".equals(part1)){
            part1 = "00";
            time2 = part1.concat(":").concat(part2);
        }else time2 = part1.concat(":").concat(part2);
    }
}

```

```

time2=time2.replace("PM", "");
time2=time2.replace("AM", "");
time1=time1.concat(":00");
time2=time2.concat(":00");
DateFormat dateFormat1 = new SimpleDateFormat("yyyy-MM-dd ");
String strDate1 = dateFormat1.format(jDateChooser1.getDate());
String date1 = strDate1.concat(time1);
DateFormat dateFormat2 = new SimpleDateFormat("yyyy-MM-dd ");
String strDate2 = dateFormat2.format(jDateChooser2.getDate());
String date2 = strDate2.concat(time2);
t1 = java.sql.Timestamp.valueOf(date1);
t2 = java.sql.Timestamp.valueOf(date2);
}

public void updateTable(){
    try{
        Class.forName("com.mysql.cj.jdbc.Driver");
        Connection con = DriverManager.getConnection("jdbc:mysql://localhost:3306/travel_agency?zeroDateTimeBehavior=CONVERT_TO_NULL", "root", "root");
        String select="SELECT * FROM offers;";
        Statement slct = con.createStatement();
        ResultSet rs = slct.executeQuery(select);
        DefaultTableModel tbModel= (DefaultTableModel) OffersTable.getModel();
        tbModel.setNumRows(0);
        while(rs.next()){
            String code = rs.getString("off_code");
            String trs = rs.getString("tr_start");
            String tre = rs.getString("tr_end");
            String costs = rs.getString("cost");
            String des = rs.getString("dest_id");
            String tb_data[] = {code, trs, tre, costs, des};
            tbModel.addRow(tb_data);
        }
        con.close();
    }catch(ClassNotFoundException | SQLException e){JOptionPane.showMessageDialog(this, e.getMessage());}
}

public void updateCombo(){
    PickTime1.setText("Set Time");
    PickTime2.setText("Set Time");
    try{
        Class.forName("com.mysql.cj.jdbc.Driver");
        Connection con = DriverManager.getConnection("jdbc:mysql://localhost:3306/travel_agency?zeroDateTimeBehavior=CONVERT_TO_NULL", "root", "root");
        String select="SELECT dst_id FROM destination;";
        Statement slct = con.createStatement();
        ResultSet rs = slct.executeQuery(select);
        DefaultComboBoxModel mod = new DefaultComboBoxModel();
        mod.removeAllElements();
        while(rs.next()){
            int box = rs.getInt("dst_id");
            mod.addElement(box);
        }
        dest_id.setModel(mod);
        con.close();
    }catch(ClassNotFoundException | SQLException e){JOptionPane.showMessageDialog(this, e.getMessage());}
}

private void PickTime2MouseClicked(java.awt.event.MouseEvent evt) {
    timePicker2.showPopup(this, 100, 100);
    String time = timePicker2.getSelectedTime();
    time=time.replace("MP", "PM");
    PickTime2.setText(time);
}

private void PickTime1MouseClicked(java.awt.event.MouseEvent evt) {
    timePicker1.showPopup(this, 100, 100);
    String time = timePicker1.getSelectedTime();
    time=time.replace("MP", "PM");
    PickTime1.setText(time);
}

private void formWindowOpened(java.awt.event.WindowEvent evt) {
    updateCombo();
    updateTable();
}

private void OffersTableMouseClicked(java.awt.event.MouseEvent evt) {
    DefaultTableModel tbModel= (DefaultTableModel) OffersTable.getModel();
    String start = tbModel.getValueAt(OffersTable.getSelectedRow(), 1).toString();
    String end = tbModel.getValueAt(OffersTable.getSelectedRow(), 2).toString();
}

```

```

String costs = tbModel.getValueAt(OffersTable.getSelectedRow(), 3).toString();
String dest = tbModel.getValueAt(OffersTable.getSelectedRow(), 4).toString();

cost.setText(costs);
dest_id.setSelectedItem(Integer.valueOf(dest));
String[] parts1 = start.split(" ");
String part1s = parts1[0];
String part2s = parts1[1];

jDateChooser1.setDate(java.sql.Date.valueOf(part1s));
PickTime1.setText(part2s);

String[] parts2 = end.split(" ");
String part1e = parts2[0];
String part2e = parts2[1];
jDateChooser2.setDate(java.sql.Date.valueOf(part1e));
PickTime2.setText(part2e);
}

private void InsertActionPerformed(java.awt.event.ActionEvent evt) {
    try{
        Class.forName("com.mysql.cj.jdbc.Driver");
        Connection con = DriverManager.getConnection("jdbc:mysql://localhost:3306/travel_agency?zeroDateTimeBehavior=CONVERT_TO_NULL", "root", "root");
        String insert="INSERT INTO offers(off_code,tr_start,tr_end,cost,dest_id) VALUES(null,?,?,?,?)";
        PreparedStatement insrt = con.prepareStatement(insert);
        convertTimeStamp();
        insrt.setTimestamp(1,t1);
        insrt.setTimestamp(2,t2);
        insrt.setFloat(3,Float.parseFloat(cost.getText()));
        insrt.setInt(4, Integer.parseInt(dest_id.getSelectedItem().toString()));
        insrt.execute();
        updateTable();
        JOptionPane.showMessageDialog(this, "Inserted Succesfully");
    }catch(ClassNotFoundException | SQLException e){JOptionPane.showMessageDialog(this, e.getMessage());}
}

private void Delete1ActionPerformed(java.awt.event.ActionEvent evt) {
    try{
        Class.forName("com.mysql.cj.jdbc.Driver");
        DefaultTableModel tbModel= (DefaultTableModel) OffersTable.getModel();
        Connection con = DriverManager.getConnection("jdbc:mysql://localhost:3306/travel_agency?zeroDateTimeBehavior=CONVERT_TO_NULL", "root", "root");
        String delete="DELETE FROM offers WHERE off_code = ?";
        PreparedStatement dlt = con.prepareStatement(delete);
        dlt.setInt(1, Integer.parseInt(tbModel.getValueAt(OffersTable.getSelectedRow(), 0).toString()));
        dlt.executeUpdate();
        updateTable();
        jDateChooser1.setDate(null);
        jDateChooser2.setDate(null);
        PickTime1.setText("");
        PickTime2.setText("");
        cost.setText("");
        JOptionPane.showMessageDialog(this, "Deleted Succesfully");
        con.close();
    }catch(ClassNotFoundException | SQLException e){JOptionPane.showMessageDialog(this, e.getMessage());}
}

private void Update1ActionPerformed(java.awt.event.ActionEvent evt) {
    convertTimeStamp();
    try{
        Class.forName("com.mysql.cj.jdbc.Driver");
        DefaultTableModel tbModel= (DefaultTableModel) OffersTable.getModel();
        Connection con = DriverManager.getConnection("jdbc:mysql://localhost:3306/travel_agency?zeroDateTimeBehavior=CONVERT_TO_NULL", "root", "root");
        String update="UPDATE offers SET tr_start = ?, tr_end = ?, cost = ?, dest_id = ? WHERE off_code = ?";
        PreparedStatement upd = con.prepareStatement(update);
        upd.setTimestamp(1, t1);
        upd.setTimestamp(2, t2);
        upd.setFloat(3, Float.parseFloat(cost.getText()));
        upd.setInt(4, Integer.parseInt(dest_id.getSelectedItem().toString()));
        upd.setInt(5, Integer.parseInt(tbModel.getValueAt(OffersTable.getSelectedRow(), 0).toString()));
        upd.executeUpdate();
        updateTable();
        JOptionPane.showMessageDialog(this, "Updated Succesfully");
        con.close();
    }catch(ClassNotFoundException | SQLException e){JOptionPane.showMessageDialog(this, e.getMessage());}
}

private void CancelActionPerformed(java.awt.event.ActionEvent evt) {

```



```

dispose();
Main main = new Main();
main.show();
}

```

## phones.java

### Κώδικας:

```

public void updateTable(){
    try{
        Class.forName("com.mysql.cj.jdbc.Driver");
        Connection con = DriverManager.getConnection("jdbc:mysql://localhost:3306/travel_agency?zeroDateTimeBehavior=CONVERT_TO_NULL", "root", "root");
        String select="SELECT * FROM phones;";
        Statement slct = con.createStatement();
        ResultSet rs = slct.executeQuery(select);
        DefaultTableModel tbModel= (DefaultTableModel) PhonesTable.getModel();
        tbModel.setNumRows(0);
        while(rs.next()){
            String code = rs.getString("ph_br_code");
            String number = rs.getString("ph_number");
            String tb_data[] = {code,number};
            tbModel.addRow(tb_data);
        }
        con.close();
    }catch(ClassNotFoundException | SQLException e){JOptionPane.showMessageDialog(this, e.getMessage());}
}

public void updateCombo(){
    try{
        Class.forName("com.mysql.cj.jdbc.Driver");
        Connection con = DriverManager.getConnection("jdbc:mysql://localhost:3306/travel_agency?zeroDateTimeBehavior=CONVERT_TO_NULL", "root", "root");
        String select="SELECT br_code FROM branch;";
        Statement slct = con.createStatement();
        ResultSet rs = slct.executeQuery(select);
        DefaultComboBoxModel mod = new DefaultComboBoxModel();
        mod.removeAllElements();
        while(rs.next()){
            String box = "Branch: "+rs.getInt("br_code")+"";
            mod.addElement(box);
        }
        br_code.setModel(mod);
        con.close();
    }catch(ClassNotFoundException | SQLException e){JOptionPane.showMessageDialog(this, e.getMessage());}
}

private void formWindowOpened(java.awt.event.WindowEvent evt) {
    updateCombo();
    updateTable();
}

private void PhonesTableMouseClicked(java.awt.event.MouseEvent evt) {
    DefaultTableModel tbModel= (DefaultTableModel) PhonesTable.getModel();
    String code = "Branch: "+tbModel.getValueAt(PhonesTable.getSelectedRow(), 0).toString()+"";
    String number = tbModel.getValueAt(PhonesTable.getSelectedRow(), 1).toString();
    br_code.setSelectedItem(code);
    ph_number.setText(number);
}

private void InsertActionPerformed(java.awt.event.ActionEvent evt) {
    try{
        Class.forName("com.mysql.cj.jdbc.Driver");
        Connection con = DriverManager.getConnection("jdbc:mysql://localhost:3306/travel_agency?zeroDateTimeBehavior=CONVERT_TO_NULL", "root", "root");
        String insert="INSERT INTO phones(ph_br_code,ph_number) VALUES(?,?)";
        PreparedStatement insrt = con.prepareStatement(insert);
        String code = br_code.getSelectedItem().toString().replaceAll("[^0-9]", "");
        int i=Integer.parseInt(code);
        insrt.setInt(1,i);
        insrt.setString(2,ph_number.getText());
        insrt.execute();
        updateTable();
        JOptionPane.showMessageDialog(this, "Inserted Succesfully");
    }catch(ClassNotFoundException | SQLException e){JOptionPane.showMessageDialog(this, e.getMessage());}
}

private void Delete1ActionPerformed(java.awt.event.ActionEvent evt) {
    try{
        Class.forName("com.mysql.cj.jdbc.Driver");

```

```

Connection con = DriverManager.getConnection("jdbc:mysql://localhost:3306/travel_agency?zeroDateTimeBehavior=CONVERT_TO_NULL", "root", "root");
String delete="DELETE FROM phones WHERE ph_br_code = ? AND ph_number = ?";
PreparedStatement dlt = con.prepareStatement(delete);
String code = br_code.getSelectedItem().toString().replaceAll("[^0-9]", "");
int i=Integer.parseInt(code);
dlt.setInt(1,i);
dlt.setString(2,ph_number.getText());
dlt.execute();
ph_number.setText("");
updateTable();
JOptionPane.showMessageDialog(this, "Deleted Succesfully");
}catch(ClassNotFoundException | SQLException e){JOptionPane.showMessageDialog(this, e.getMessage());}
}

private void Update1ActionPerformed(java.awt.event.ActionEvent evt) {
JOptionPane.showMessageDialog(this, "You cannot Update this Table! Try Using Insert/Delete Instead.");
}

private void CancelActionPerformed(java.awt.event.ActionEvent evt) {
dispose();
Main main = new Main();
main.show();
}

```

[rsv.java](#)

## Κώδικας:

```

public void updateTable(){
    try{
        Class.forName("com.mysql.cj.jdbc.Driver");
        Connection con = DriverManager.getConnection("jdbc:mysql://localhost:3306/travel_agency?zeroDateTimeBehavior=CONVERT_TO_NULL", "root", "root");
        String select="SELECT * FROM reservation;";
        Statement slct = con.createStatement();
        ResultSet rs = slct.executeQuery(select);
        DefaultTableModel tbModel= (DefaultTableModel) ReservationTable.getModel();
        tbModel.setNumRows(0);
        while(rs.next()){
            String id = rs.getString("res_tr_id");
            String seatnum = rs.getString("res_seatnum");
            String name = rs.getString("res_name");
            String lname = rs.getString("res_lname");
            String isadult = rs.getString("res_isadult");
            String tb_data[] = {id,seatnum,name,lname,isadult};
            tbModel.addRow(tb_data);
        }
        con.close();
    }catch(ClassNotFoundException | SQLException e){JOptionPane.showMessageDialog(this, e.getMessage());}
}

public void updateCombo(){
    try{
        Class.forName("com.mysql.cj.jdbc.Driver");
        Connection con = DriverManager.getConnection("jdbc:mysql://localhost:3306/travel_agency?zeroDateTimeBehavior=CONVERT_TO_NULL", "root", "root");
        String select="SELECT tr_id FROM trip ORDER BY tr_id;";
        Statement slct = con.createStatement();
        ResultSet rs = slct.executeQuery(select);
        DefaultComboBoxModel mod = new DefaultComboBoxModel();
        mod.removeAllElements();
        while(rs.next()){mod.addElement("Trip: "+rs.getInt("tr_id")+"");}
        res_tr_id.setModel(mod);
        con.close();
    }catch(ClassNotFoundException | SQLException e){JOptionPane.showMessageDialog(this, e.getMessage());}
}

private void formWindowOpened(java.awt.event.WindowEvent evt) {
    updateCombo();
    updateTable();
}

private void ReservationTableMouseClicked(java.awt.event.MouseEvent evt) {
    DefaultTableModel tbModel= (DefaultTableModel) ReservationTable.getModel();
    String id = "Trip: "+tbModel.getValueAt(ReservationTable.getSelectedRow(), 0).toString()+"";
    String seat = tbModel.getValueAt(ReservationTable.getSelectedRow(), 1).toString();
    String name = tbModel.getValueAt(ReservationTable.getSelectedRow(), 2).toString();
    String lname = tbModel.getValueAt(ReservationTable.getSelectedRow(), 3).toString();
    String isadult = tbModel.getValueAt(ReservationTable.getSelectedRow(), 4).toString();
    res_tr_id.setSelectedItem(id);
    res_seatnum.setText(seat);
}

```

```

        res_name.setText(name);
        res_lname.setText(lname);
        res_isadult.setSelectedItem(isadult);
        seatnum = Integer.parseInt(res_seatnum.getText());
    }

private void InsertActionPerformed(java.awt.event.ActionEvent evt) {
    try{
        Class.forName("com.mysql.cj.jdbc.Driver");
        Connection con = DriverManager.getConnection("jdbc:mysql://localhost:3306/travel_agency?zeroDateTimeBehavior=CONVERT_TO_NULL", "root", "root");
        String insert="INSERT INTO reservation(res_tr_id,res_seatnum,res_name,res_lname,res_isadult) VALUES(?,?,?,?,?)";
        PreparedStatement insrt = con.prepareStatement(insert);
        String code = res_tr_id.getSelectedItem().toString().replaceAll("[^0-9]", "");
        int i=Integer.parseInt(code);
        insrt.setInt(1,i);
        insrt.setInt(2,Integer.parseInt(res_seatnum.getText()));
        insrt.setString(3,res_name.getText());
        insrt.setString(4,res_lname.getText());
        insrt.setString(5,res_isadult.getSelectedItem().toString());
        insrt.execute();
        updateTable();
        JOptionPane.showMessageDialog(this, "Inserted Succesfully");
    }catch(ClassNotFoundException | SQLException e){JOptionPane.showMessageDialog(this, e.getMessage());}
}

private void Delete1ActionPerformed(java.awt.event.ActionEvent evt) {
    try{
        Class.forName("com.mysql.cj.jdbc.Driver");
        Connection con = DriverManager.getConnection("jdbc:mysql://localhost:3306/travel_agency?zeroDateTimeBehavior=CONVERT_TO_NULL", "root", "root");
        String delete="DELETE FROM reservation WHERE res_tr_id = ? AND res_seatnum = ?";
        PreparedStatement dlt = con.prepareStatement(delete);
        int i = Integer.parseInt(res_tr_id.getSelectedItem().toString().replaceAll("[^0-9]", ""));
        dlt.setInt(1, i);
        dlt.setInt(2,Integer.parseInt(res_seatnum.getText()));
        dlt.executeUpdate();
        updateTable();
        res_seatnum.setText("");
        res_name.setText("");
        res_lname.setText("");
        JOptionPane.showMessageDialog(this, "Deleted Succesfully");
        con.close();
    }catch(ClassNotFoundException | SQLException e){JOptionPane.showMessageDialog(this, e.getMessage());}
}

private void Update1ActionPerformed(java.awt.event.ActionEvent evt) {
    if(seatnum == Integer.parseInt(res_seatnum.getText())){
        try{
            Class.forName("com.mysql.cj.jdbc.Driver");
            Connection con = DriverManager.getConnection("jdbc:mysql://localhost:3306/travel_agency?zeroDateTimeBehavior=CONVERT_TO_NULL", "root", "root");
            String update="UPDATE reservation SET res_name = ?, res_lname = ?, res_isadult = ? WHERE res_tr_id = ? AND res_seatnum = ?";
            PreparedStatement upd = con.prepareStatement(update);
            int i = Integer.parseInt(res_tr_id.getSelectedItem().toString().replaceAll("[^0-9]", ""));
            upd.setString(1,res_name.getText());
            upd.setString(2,res_lname.getText());
            upd.setString(3,res_isadult.getSelectedItem().toString());
            upd.setInt(4, i);
            upd.setInt(5,seatnum);
            upd.executeUpdate();
            updateTable();
            JOptionPane.showMessageDialog(this, "Updated Succesfully");
            con.close();
        }catch(ClassNotFoundException | SQLException e){JOptionPane.showMessageDialog(this, e.getMessage());}
    }else{JOptionPane.showMessageDialog(this, "You cannot Update the Seat Number! Try Using Insert/Delete Instead.");}
}

private void CancelActionPerformed(java.awt.event.ActionEvent evt) {
    dispose();
    Main main = new Main();
    main.show();
}

```

## Κώδικας:

```
static Timestamp t1;
static Timestamp t2;
static int temp;
public void convertTimeStamp(){
    String time1 = PickTime1.getText();
    String time2 = PickTime2.getText();
    time1=time1.replace(" ", "");
    time2=time2.replace(" ", "");
    if(time1.contains("PM")){
        String[] parts = time1.split(":");
        String part1 = parts[0];
        String part2 = parts[1];
        if(!"12".equals(part1)){
            int part1int = Integer.parseInt(part1)+12;
            part1 = Integer.toString(part1int);
            time1 = part1.concat(":").concat(part2);
        }else time1 = part1.concat(":").concat(part2);
    }else if(time1.contains("AM")){
        String[] parts = time1.split(":");
        String part1 = parts[0];
        String part2 = parts[1];
        if("12".equals(part1)){
            part1 = "00";
            time1 = part1.concat(":").concat(part2);
        }else time1 = part1.concat(":").concat(part2);
    }else{
        String[] parts = time1.split(":");
        String part1 = parts[0];
        String part2 = parts[1];
        time1 = part1.concat(":").concat(part2);
    }
    time1=time1.replace("PM", "");
    time1=time1.replace("AM", "");

    if(time2.contains("PM")){
        String[] parts = time2.split(":");
        String part1 = parts[0];
        String part2 = parts[1];
        if(!"12".equals(part1)){
            int part1int = Integer.parseInt(part1)+12;
            part1 = Integer.toString(part1int);
            time2 = part1.concat(":").concat(part2);
        }else time2 = part1.concat(":").concat(part2);
    }else{
        String[] parts = time2.split(":");
        String part1 = parts[0];
        String part2 = parts[1];
        if("12".equals(part1)){
            part1 = "00";
            time2 = part1.concat(":").concat(part2);
        }else time2 = part1.concat(":").concat(part2);
    }
    time2=time2.replace("PM", "");
    time2=time2.replace("AM", "");
    time1=time1.concat(":00");
    time2=time2.concat(":00");
    DateFormat dateFormat1 = new SimpleDateFormat("yyyy-MM-dd ");
    String strDate1 = dateFormat1.format(jDateChooser1.getDate());
    String date1 = strDate1.concat(time1);
    DateFormat dateFormat2 = new SimpleDateFormat("yyyy-MM-dd ");
    String strDate2 = dateFormat2.format(jDateChooser2.getDate());
    String date2 = strDate2.concat(time2);
    t1 = java.sql.Timestamp.valueOf(date1);
    t2 = java.sql.Timestamp.valueOf(date2);
}

public void updateTable(){
    try{
        Class.forName("com.mysql.cj.jdbc.Driver");
        Connection con = DriverManager.getConnection("jdbc:mysql://localhost:3306/travel_agency?zeroDateTimeBehavior=CONVERT_TO_NULL", "root", "root");
        String select="SELECT * FROM travel_to;";
        Statement slct = con.createStatement();
        ResultSet rs = slct.executeQuery(select);
    }
}
```

```

        DefaultTableModel tbModel= (DefaultTableModel) TravelToTable.getModel();
        tbModel.setNumRows(0);
        while(rs.next()){
            String trid = rs.getString("to_tr_id");
            String dstid = rs.getString("to_dst_id");
            String arr = rs.getString("to_arrival");
            String dep = rs.getString("to_departure");
            String tb_data[]={trid, dstid, arr, dep};
            tbModel.addRow(tb_data);
        }
        con.close();
    }catch(ClassNotFoundException | SQLException e){JOptionPane.showMessageDialog(this, e.getMessage());}
}

public void updateCombo(){
    PickTime1.setText("Set Time");
    PickTime2.setText("Set Time");
    try{
        Class.forName("com.mysql.cj.jdbc.Driver");
        Connection con = DriverManager.getConnection("jdbc:mysql://localhost:3306/travel_agency?zeroDateTimeBehavior=CONVERT_TO_NULL", "root", "root");
        String select1="SELECT dst_id FROM destination;";
        String select2="SELECT tr_id FROM trip ORDER BY tr_id;";
        Statement slct1 = con.createStatement();
        Statement slct2 = con.createStatement();
        ResultSet rs1 = slct1.executeQuery(select1);
        ResultSet rs2 = slct2.executeQuery(select2);
        DefaultComboBoxModel mod1 = new DefaultComboBoxModel();
        DefaultComboBoxModel mod2 = new DefaultComboBoxModel();
        mod1.removeAllElements();
        mod2.removeAllElements();
        while(rs1.next()){
            int box1 = rs1.getInt("dst_id");
            mod1.addElement(box1);
        }
        while(rs2.next()){
            int box2 = rs2.getInt("tr_id");
            mod2.addElement(box2);
        }
        to_dst_id.setModel(mod1);
        to_tr_id.setModel(mod2);
        con.close();
    }catch(ClassNotFoundException | SQLException e){JOptionPane.showMessageDialog(this, e.getMessage());}
}

private void PickTime2MouseClicked(java.awt.event.MouseEvent evt) {
    timePicker2.showPopup(this, 100, 100);
    String time = timePicker2.getSelectedTime();
    time=time.replace("MP", "PM");
    PickTime2.setText(time);
}

private void PickTime1MouseClicked(java.awt.event.MouseEvent evt) {
    timePicker1.showPopup(this, 100, 100);
    String time = timePicker1.getSelectedTime();
    time=time.replace("MP", "PM");
    PickTime1.setText(time);
}

private void formWindowOpened(java.awt.event.WindowEvent evt) {
    updateTable();
    updateCombo();
}

private void TravelToTableMouseClicked(java.awt.event.MouseEvent evt) {
    DefaultTableModel tbModel= (DefaultTableModel) TravelToTable.getModel();
    String trid = tbModel.getValueAt(TravelToTable.getSelectedRow(), 0).toString();
    String dstid = tbModel.getValueAt(TravelToTable.getSelectedRow(), 1).toString();
    String arr = tbModel.getValueAt(TravelToTable.getSelectedRow(), 2).toString();
    String dep = tbModel.getValueAt(TravelToTable.getSelectedRow(), 3).toString();

    to_tr_id.setSelectedItem(Integer.valueOf(trid));
    to_dst_id.setSelectedItem(Integer.valueOf(dstid));
    temp = Integer.parseInt(tbModel.getValueAt(TravelToTable.getSelectedRow(), 1).toString());

    String[] parts1 = arr.split(" ");
    String part1s = parts1[0];
    String part2s = parts1[1];

    jDateChooser1.setDate(java.sql.Date.valueOf(part1s));

```

```

PickTime1.setText(part2s);

String[] parts2 = dep.split(" ");
String part1e = parts2[0];
String part2e = parts2[1];
jDateChooser2.setDate(java.sql.Date.valueOf(part1e));
PickTime2.setText(part2e);
}

private void InsertActionPerformed(java.awt.event.ActionEvent evt) {
    try{
        Class.forName("com.mysql.cj.jdbc.Driver");
        Connection con = DriverManager.getConnection("jdbc:mysql://localhost:3306/travel_agency?zeroDateTimeBehavior=CONVERT_TO_NULL", "root", "root");
        String insert="INSERT INTO travel_to(to_tr_id,to_dst_id,to_arrival,to_departure) VALUES(?,?,?,?)";
        PreparedStatement insrt = con.prepareStatement(insert);
        convertTimeStamp();
        insrt.setInt(1,Integer.parseInt(to_tr_id.getSelectedItemAt().toString()));
        insrt.setInt(2, Integer.parseInt(to_dst_id.getSelectedItemAt().toString()));
        insrt.setTimestamp(3,t1);
        insrt.setTimestamp(4,t2);
        insrt.execute();
        updateTable();
        JOptionPane.showMessageDialog(this, "Inserted Successfully");
    }catch(ClassNotFoundException | SQLException e){JOptionPane.showMessageDialog(this, e.getMessage());}
}

private void Delete1ActionPerformed(java.awt.event.ActionEvent evt) {
    try{
        Class.forName("com.mysql.cj.jdbc.Driver");
        Connection con = DriverManager.getConnection("jdbc:mysql://localhost:3306/travel_agency?zeroDateTimeBehavior=CONVERT_TO_NULL", "root", "root");
        String delete="DELETE FROM travel_to WHERE to_tr_id = ? AND to_dst_id = ?";
        PreparedStatement dlt = con.prepareStatement(delete);
        dlt.setInt(1, Integer.parseInt(to_tr_id.getSelectedItemAt().toString()));
        dlt.setInt(2, Integer.parseInt(to_dst_id.getSelectedItemAt().toString()));
        dlt.executeUpdate();
        updateTable();
        jDateChooser1.setDate(null);
        jDateChooser2.setDate(null);
        PickTime1.setText("");
        PickTime2.setText("");
        JOptionPane.showMessageDialog(this, "Deleted Successfully");
        con.close();
    }catch(ClassNotFoundException | SQLException e){JOptionPane.showMessageDialog(this, e.getMessage());}
}

private void Update1ActionPerformed(java.awt.event.ActionEvent evt) {
    convertTimeStamp();
    if(temp == Integer.parseInt(to_dst_id.getSelectedItemAt().toString())){
        try{
            Class.forName("com.mysql.cj.jdbc.Driver");
            Connection con = DriverManager.getConnection("jdbc:mysql://localhost:3306/travel_agency?zeroDateTimeBehavior=CONVERT_TO_NULL", "root", "root");
            String update="UPDATE travel_to SET to_arrival = ?, to_departure = ? WHERE to_tr_id = ? AND to_dst_id = ?";
            PreparedStatement upd = con.prepareStatement(update);
            upd.setTimestamp(1, t1);
            upd.setTimestamp(2, t2);
            upd.setInt(3, Integer.parseInt(to_tr_id.getSelectedItemAt().toString()));
            upd.setInt(4, Integer.parseInt(to_dst_id.getSelectedItemAt().toString()));
            upd.executeUpdate();
            updateTable();
            JOptionPane.showMessageDialog(this, "Updated Successfully");
            con.close();
        }catch(ClassNotFoundException | SQLException e){JOptionPane.showMessageDialog(this, e.getMessage());}
    }else{JOptionPane.showMessageDialog(this, "You cannot Update the Travel_to destination ID! Try Using Insert/Delete Instead.");}
}

private void CancelActionPerformed(java.awt.event.ActionEvent evt) {
    dispose();
    Main main = new Main();
    main.show();
}

```

## Κώδικας:

```

public void updateTable(){
    try{
        Class.forName("com.mysql.cj.jdbc.Driver");
        Connection con = DriverManager.getConnection("jdbc:mysql://localhost:3306/travel_agency?zeroDateTimeBehavior=CONVERT_TO_NULL", "root", "root");
        String select="SELECT * FROM worker;";
        Statement slct = con.createStatement();
        ResultSet rs = slct.executeQuery(select);
        DefaultTableModel tbModel= (DefaultTableModel) WorkerTable.getModel();
        tbModel.setNumRows(0);
        while(rs.next()){
            String at = rs.getString("wrk_AT");
            String name = rs.getString("wrk_name");
            String lname = rs.getString("wrk_lname");
            String salary = rs.getString("wrk_salary");
            String code = Integer.toString(rs.getInt("wrk_br_code"));
            String tb_data[] = {at, name, lname, salary, code};
            tbModel.addRow(tb_data);
        }
        con.close();
    }catch(ClassNotFoundException | SQLException e){JOptionPane.showMessageDialog(this, e.getMessage());}
}

public void updateCombo(){
    try{
        Class.forName("com.mysql.cj.jdbc.Driver");
        Connection con = DriverManager.getConnection("jdbc:mysql://localhost:3306/travel_agency?zeroDateTimeBehavior=CONVERT_TO_NULL", "root", "root");
        String select="SELECT br_code FROM branch;";
        Statement slct = con.createStatement();
        ResultSet rs = slct.executeQuery(select);
        DefaultComboBoxModel mod = new DefaultComboBoxModel();
        mod.removeAllElements();
        while(rs.next()){
            String box = "Branch: "+rs.getInt("br_code")+"";
            mod.addElement(box);
        }
        wrk_br_code.setModel(mod);
        con.close();
    }catch(ClassNotFoundException | SQLException e){JOptionPane.showMessageDialog(this, e.getMessage());}
}

private void formWindowOpened(java.awt.event.WindowEvent evt) {
    updateTable();
    updateCombo();
}

private void WorkerTableMouseClicked(java.awt.event.MouseEvent evt) {
    DefaultTableModel tbModel= (DefaultTableModel) WorkerTable.getModel();
    String at = tbModel.getValueAt(WorkerTable.getSelectedRow(), 0).toString();
    String name = tbModel.getValueAt(WorkerTable.getSelectedRow(), 1).toString();
    String lname = tbModel.getValueAt(WorkerTable.getSelectedRow(), 2).toString();
    String salary = tbModel.getValueAt(WorkerTable.getSelectedRow(), 3).toString();
    String code = "Branch: "+tbModel.getValueAt(WorkerTable.getSelectedRow(), 4).toString()+"";
    wrk_AT.setText(at);
    wrk_name.setText(name);
    wrk_lname.setText(lname);
    wrk_salary.setText(salary);
    wrk_br_code.setSelectedItem(code);
}

private void Delete1ActionPerformed(java.awt.event.ActionEvent evt) {
    try{
        Class.forName("com.mysql.cj.jdbc.Driver");
        Connection con = DriverManager.getConnection("jdbc:mysql://localhost:3306/travel_agency?zeroDateTimeBehavior=CONVERT_TO_NULL", "root", "root");
        String del= "DELETE FROM worker WHERE wrk_AT = ?;";
        PreparedStatement dlt = con.prepareStatement(del);
        dlt.setString(1, wrk_AT.getText());
        dlt.executeUpdate();
        updateTable();
        wrk_AT.setText("");
        wrk_name.setText("");
        wrk_lname.setText("");
        wrk_salary.setText("");
        JOptionPane.showMessageDialog(this, "Deleted Succesfully");
        con.close();
    }catch(ClassNotFoundException | SQLException e){JOptionPane.showMessageDialog(this, e.getMessage());}
}

```

```

}

private void Update1ActionPerformed(java.awt.event.ActionEvent evt) {
    try{
        DefaultTableModel tbModel= (DefaultTableModel) WorkerTable.getModel();
        Class.forName("com.mysql.cj.jdbc.Driver");
        Connection con = DriverManager.getConnection("jdbc:mysql://localhost:3306/travel_agency?zeroDateTimeBehavior=CONVERT_TO_NULL", "root", "root");
        String update="UPDATE worker SET wrk_name = ?, wrk_lname = ?, wrk_salary = ?, wrk_br_code = ? WHERE wrk_AT = ?";
        PreparedStatement upd = con.prepareStatement(update);
        String code = wrk_br_code.getSelectedItemId().toString().replaceAll("[^0-9]", "");
        int i=Integer.parseInt(code);
        upd.setString(1, wrk_name.getText());
        upd.setString(2, wrk_lname.getText());
        upd.setFloat(3, Float.parseFloat(wrk_salary.getText()));
        upd.setInt(4, i);
        upd.setString(5, tbModel.getValueAt(WorkerTable.getSelectedRow(), 0).toString());
        upd.executeUpdate();
        updateTable();
        JOptionPane.showMessageDialog(this, "Updated Succesfully");
        con.close();
    }catch(ClassNotFoundException | SQLException e){JOptionPane.showMessageDialog(this, e.getMessage());}
}

private void CancelActionPerformed(java.awt.event.ActionEvent evt) {
    dispose();
    Main main = new Main();
    main.show();
}

private void InsertActionPerformed(java.awt.event.ActionEvent evt) {
    try{
        Class.forName("com.mysql.cj.jdbc.Driver");
        Connection con = DriverManager.getConnection("jdbc:mysql://localhost:3306/travel_agency?zeroDateTimeBehavior=CONVERT_TO_NULL", "root", "root");
        String insert="INSERT INTO worker(wrk_AT,wrk_name,wrk_lname,wrk_salary,wrk_br_code) VALUES(?,?,?,?,?)";
        PreparedStatement insrt = con.prepareStatement(insert);
        String code = wrk_br_code.getSelectedItemId().toString().replaceAll("[^0-9]", "");
        int i=Integer.parseInt(code);
        insrt.setString(1,wrk_AT.getText());
        insrt.setString(2,wrk_name.getText());
        insrt.setString(3,wrk_lname.getText());
        insrt.setFloat(4, Float.parseFloat(wrk_salary.getText()));
        insrt.setInt(5,i);
        insrt.execute();
        updateTable();
        JOptionPane.showMessageDialog(this, "Inserted Succesfully");
    }catch(ClassNotFoundException | SQLException e){JOptionPane.showMessageDialog(this, e.getMessage());}
}

```

## Επεξήγηση Κώδικα:

Μετά την επιλογή του επιθυμητού πίνακα από τον IT Manager, ανοίγει το καινούριο JFrame. Για κάθε JFrame, χρησιμοποιούμε το ίδιο μοντέλο κώδικα. Για κάθε πεδίο του επιλεγμένου πίνακα εμφανίζονται αντίστοιχα πεδία, με τα οποία ο IT Manager μπορεί να προβάλει, τροποποιήσει ή εισάγει δεδομένα.

- ❖ Insert: Όταν ο χρήστης επιθυμεί να κάνει Insert στον επιλεγμένο πίνακα, αφού έχει συμπληρώσει τα διαθέσιμα πεδία, με ένα try-catch block γίνεται η σύνδεση στην βάση δεδομένων. Έπειτα αρχικοποιούμε ένα preparedStatement το οποίο παίρνει σαν όρισμα ένα String της παρακάτω μορφής, που σε κάθε περίπτωση πρέπει να αποτελείται από τόσα ερωτηματικά όσες και οι τιμές των πεδίων του πίνακα.

```
String insert="INSERT INTO branch(br_code,br_street,br_num,br_city) VALUES(null,?,?,?)";
```

Έπειτα με συναρτήσεις setString/Int/Float παίρνουμε τα δεδομένα από τα πεδία που συμπλήρωσε ο IT Manager και κάνουμε execute() το preparedStatement, εισάγοντας τα δεδομένα στον πίνακα, ενώ ταυτόχρονα εμφανίζεται αντίστοιχο μήνυμα επιβεβαίωσης ή λάθους (Σε περίπτωση Trigger ή exception με JOptionPane). Τέλος, με την συνάρτηση updateTable() (η επεξήγηση της οποίας θα γίνει μετά την επεξήγηση του delete) ενημερώνουμε το jTable που εμφανίζεται στον χρήστη με τα στοιχεία του κάθε πίνακα.

- ❖ Update: Όταν ο χρήστης επιθυμεί να κάνει Update στον επιλεγμένο πίνακα, αφού έχει αλλάξει τα διαθέσιμα πεδία, με ένα try-catch block γίνεται η σύνδεση στην βάση δεδομένων. Έπειτα αρχικοποιούμε ένα preparedStatement το οποίο παίρνει σαν όρισμα ένα String της παρακάτω μορφής, που σε κάθε περίπτωση πρέπει να αποτελείται από τόσα ερωτηματικά όσες και οι τιμές των πεδίων του πίνακα.

```
String update="UPDATE branch SET br_street = ?, br_num = ?, br_city = ? WHERE br_code = ?";
```



Έπειτα με συναρτήσεις `setString/Int/Float` παίρνουμε τα δεδομένα από τα πεδία που συμπλήρωσε ο IT Manager και κάνουμε `executeUpdate()` το `preparedStatement`, ενημερώνοντας τα δεδομένα στον πίνακα, ενώ ταυτόχρονα εμφανίζεται αντίστοιχο μήνυμα επιβεβαίωσης ή λάθους (Σε περίπτωση `Trigger` ή `exception` με `JOptionPane`). Τέλος, με την συνάρτηση `updateTable()` ενημερώνουμε το `JTable` που εμφανίζεται στον χρήστη με τα στοιχεία του κάθε πίνακα.

- ❖ Delete: Όταν ο χρήστης επιθυμεί να κάνει `Delete` στον επιλεγμένο πίνακα, αφού έχει επιλέξει από το `JTable` την εγγραφή που θέλει να διαγράψει, με ένα `try-catch block` γίνεται η σύνδεση στην βάση δεδομένων. Έπειτα αρχικοποιούμε ένα `preparedStatement` το οποίο παίρνει σαν όρισμα ένα `String` της παρακάτω μορφής, που σε κάθε περίπτωση πρέπει να αποτελείται από τόσα ερωτηματικά όσα τα κλειδιά του πίνακα αυτού.

```
String del="DELETE FROM branch WHERE br_code = ?";
```

Έπειτα με συναρτήσεις `setString/Int/Float` παίρνουμε τα δεδομένα των κλειδιών από την εγγραφή που επέλεξε ο IT Manager από το `JTable` και κάνουμε `execute()` το `preparedStatement`, διαγράφοντας τα δεδομένα από τον πίνακα, ενώ ταυτόχρονα εμφανίζεται αντίστοιχο μήνυμα επιβεβαίωσης ή λάθους (Σε περίπτωση `Trigger` ή `exception` με `JOptionPane`). Τέλος, με την συνάρτηση `updateTable()` ενημερώνουμε το `JTable` που εμφανίζεται στον χρήστη με τα στοιχεία του κάθε πίνακα, καθώς επίσης «αδειάζουμε» όλα τα διαθέσιμα πεδία που έχει ο IT Manager.

- ❖ updateTable(): Είναι μία συνάρτηση που χρησιμοποιείται σε κάθε `JFrame` μετά την επιλογή ποιου πίνακα θέλει να επεξεργαστεί ο IT Manager. Είναι υπεύθυνη για την ενημέρωση του `JTable` που έχει στην διάθεσή του ο IT Manager μετά από οποιαδήποτε εισαγωγή, ενημέρωση, διαγραφή που συμβεί στον πίνακα, αλλά καθώς και την εισαγωγή των στοιχείων από την βάση δεδομένων στο `JTable` με την εμφάνιση του `JFrame`. Αποτελείται από ένα `try-catch block`, ώστε να γίνει η σύνδεση με την βάση δεδομένων και ένα `preparedStatement` το οποίο παίρνει σαν όρισμα ένα `String` της παρακάτω μορφής, που σε κάθε περίπτωση πρέπει να αποτελείται από το όνομα του πίνακα που θέλουμε να εμφανίσουμε κάθε φορά.

```
String select="SELECT * FROM branch;";
```

Τέλος, αποθηκεύουμε κάθε πεδίο κάθε εγγραφής του πίνακα σε μία μεταβλητή και προσθέτουμε μία γραμμή στο `Model` αυτού του `JTable` με αυτά τα πεδία, με τις παρακάτω εντολές:

```
String tb_data[] = {code, street, num, city};  
tbModel.addRow( rowData:tb_data );
```

- ❖ (TableName)TableMouseClicked: Είναι μία συνάρτηση, η οποία με το που πατήσει ο χρήστης σε μία εγγραφή από τον πίνακα `JTable`, ενημερώνει τα πεδία που είναι διαθέσιμα στον χρήστη με τις τιμές αυτής της εγγραφής. Συγκεκριμένα, θέτουμε σε μεταβλητές με την βοήθεια της συνάρτησης `getValueAt` τα δεδομένα από κάθε στήλη του `JTable` και έπειτα θέτουμε στα πεδία με συναρτήσεις `setText/setSelectedItem` τις τιμές των παραπάνω μεταβλητών.
- ❖ updateCombo(): Είναι μία συνάρτηση που χρησιμοποιείται σε κάθε `JFrame` που σαν πεδίο υπάρχει `JComboBox`. Είναι ιδιαίτερα χρήσιμη καθώς με αυτήν ενημερώνουμε τις διαθέσιμες τιμές του κάθε `JComboBox` δυναμικά και αποτρέπουμε τον IT Manager από το να προσπαθήσει να επιλέξει κάποια τιμή η οποία δεν υπάρχει στην βάση δεδομένων. Αποτελείται από ένα `try-catch block` το οποίο αρχικοποιεί την σύνδεση με την βάση δεδομένων και ένα `Statement`. Έπειτα δημιουργούμε ένα `ResultSet` για να πάρουμε όλες τις τιμές από το συγκεκριμένο πεδίο και πίνακα που μας ενδιαφέρει για το συγκεκριμένο `JComboBox` με ένα `String`. Τέλος, αφού έχουμε αδειάσει τα περιεχόμενα του `JComboBox` με την συνάρτηση `removeAllElements()` (από τυχόν προηγούμενες τιμές που μπορεί να μην υπάρχουν πλέον), εισάγουμε όλα τα στοιχεία που παίρνουμε από το `ResultSet` με την συνάρτηση `addElement` σε ένα `model` και θέτουμε το μοντέλο αυτό στο `JComboBox` με την συνάρτηση `setModel()`.
- ❖ convertTimeStamp(): Είναι μια συνάρτηση που χρησιμοποιείται σε κάθε `JFrame` που σαν πεδίο υπάρχει `Date/Time` επιλογή. Εδώ πρέπει να αναφέρουμε πως χρησιμοποιούμε δύο διαφορετικά `Plugins` για την επιλογή ημερομηνίας και ώρας όπου χρειάζεται. Το `Plugin` για την επιλογή ημερομηνίας επιστρέφει την ημερομηνία σε `Date` μορφή και το `Plugin` για την επιλογή ώρας σε `String`. Για αυτόν τον λόγο, πρέπει να γίνει επεξεργασία αυτών των δεδομένων και μετατροπή σε `java.sql.Timestamp`, ώστε να μπορέσουμε να τα εισάγουμε στην βάση δεδομένων. Εν συνεχεία, το `Plugin` για την επιλογή ώρας, χρησιμοποιεί μορφή 12 ωρών για την επιλογή της ώρας ενώ η `SQL` χρησιμοποιεί μορφή 24 ωρών. Επομένως, χρειάζεται και αυτού του είδους η μετατροπή. Αυτή είναι και η χρήση αυτής της συνάρτησης. Συγκεκριμένα, παίρνει την ώρα από το πεδίο που επιλέγουμε την ώρα και ανάλογα του αν υπάρχουν οι χαρακτήρες “AM” ή “PM” (που συμβολίζουν προ/μετά μεσημβρίας) τους αφαιρεί και κάνει τις ανάλογες αλλαγές χωρίζοντας το `String` της ώρας με `regex` το σύμβολο “:” έπειτα αφού μετατρέψει την μορφή 12 ωρών σε 24 ωρών, ενώνει το `String` και το θέτει σε μία μεταβλητή. Μετατρέπουμε την ημερομηνία από `Date` σε `String` με την συνάρτηση `format` και τέλος ενώνουμε την συνάρτηση `concat()` τα δύο `String`, ώστε να έχουμε ένα `String` με ημερομηνία και ώρα μαζί. Τέλος,

Θέτουμε σε μεταβλητές Timestamp t1 και t2 τα δύο String μετά από μετατροπή τους σε java.sql.Timestamp με την συνάρτηση valueOf.

- ❖ Cancel: Το κουμπί Cancel χρησιμοποιείται από τον IT Manager όταν θέλει να επιστρέψει στο αρχικό μενού. Ο κώδικας είναι αρκετά απλός, καθώς κάνει dispose() το παρόν JFrame και κάνει show() το Main.

## CheckTrip.java

### Κώδικας:

```
static java.sql.Date t1 = null;
static java.sql.Date t2 = null;

public void updateTable(){
    try{
        DefaultTableModel tbModel= (DefaultTableModel) CheckTripTable.getModel();
        String brcode = br_code.getSelectedItem().toString().replaceAll("[^0-9]", "");
        int i=Integer.parseInt(brcode);
        Class.forName("com.mysql.cj.jdbc.Driver");
        Connection con = DriverManager.getConnection("jdbc:mysql://localhost:3306/travel_agency?zeroDateTimeBehavior=CONVERT_TO_NULL", "root", "root");
        String s = "{CALL date_check('"+i+"','"+t1+"','"+t2+"')}";
        CallableStatement cs = con.prepareCall(s);
        cs.execute();
        ResultSet rs = cs.getResultSet();
        tbModel.setNumRows(0);
        while(rs.next()){
            String cost = rs.getString("trcost");
            String max = rs.getString("maxseats_");
            String res = rs.getString("reservations_");
            String seatdiff = rs.getString("seatdiff_");
            String drvn = rs.getString("drv_name_");
            String drvlN = rs.getString("drv_lname_");
            String guin = rs.getString("gui_name_");
            String guilN = rs.getString("gui_lname_");
            String dep = rs.getString("tr_dep_");
            String ret = rs.getString("tr_ret_");
            String tb_data[] = {cost, max, res, seatdiff, drvn, drvlN, guin, guilN, dep, ret};
            tbModel.addRow(tb_data);
        }
        con.close();
    }catch(ClassNotFoundException | SQLException e){JOptionPane.showMessageDialog(this, e.getMessage());}
}

public void updateCombo(){
    try{
        Class.forName("com.mysql.cj.jdbc.Driver");
        Connection con = DriverManager.getConnection("jdbc:mysql://localhost:3306/travel_agency?zeroDateTimeBehavior=CONVERT_TO_NULL", "root", "root");
        String select="SELECT br_code FROM branch;";
        Statement slct = con.createStatement();
        ResultSet rs = slct.executeQuery(select);
        DefaultComboBoxModel mod = new DefaultComboBoxModel();
        mod.removeAllElements();
        while(rs.next()){
            String box = "Branch: "+rs.getInt("br_code")+"";
            mod.addElement(box);
        }
        br_code.setModel(mod);
        con.close();
    }catch(ClassNotFoundException | SQLException e){JOptionPane.showMessageDialog(this, e.getMessage());}
}

private void formWindowOpened(java.awt.event.WindowEvent evt) {
    updateCombo();
}

private void Cancel1ActionPerformed(java.awt.event.ActionEvent evt) {
    dispose();
    Main main = new Main();
    main.show();
}

private void Search1ActionPerformed(java.awt.event.ActionEvent evt) {
    if(jDateChooser2.getDate() != null || jDateChooser1.getDate() != null){
        t1 = new java.sql.Date(jDateChooser1.getDate().getTime());
        t2 = new java.sql.Date(jDateChooser2.getDate().getTime());
        updateTable();
    }else JOptionPane.showMessageDialog(this, "Please Fill all the Fields in order to proceed!");
}
```

## Επεξήγηση:

Αυτό το JFrame χρησιμοποιείται στην περίπτωση που ο IT Manager θέλει να προβάλει τα ταξίδια που διοργανώνονται από κάποιο branch για ένα συγκεκριμένο διάστημα ημερομηνιών. Όπως και στα παραπάνω χρησιμοποιούνται συναρτήσεις όπως οι, updateCombo(), updateTable() και Cancel. Ωστόσο, σε αυτήν την περίπτωση η updateTable() είναι τροποποιημένη ώστε να καλεί το Stored Procedure date\_check. Αναλυτικότερα, αρχικοποιούμε μία σύνδεση με την βάση δεδομένων, με ένα try-catch block και δημιουργούμε ένα CallableStatement το οποίο παίρνει σαν όρισμα το παρακάτω String:

```
String s = "{CALL date_check('"+i+"', '"+t1+"', '"+t2+"')}"
```

Τέλος, αποθηκεύουμε κάθε πεδίο κάθε εγγραφής του πίνακα σε μία μεταβλητή και προσθέτουμε μία γραμμή στο Model αυτού του jTable με αυτά τα πεδία, με τις παρακάτω εντολές:

```
String tb_data[] = {cost, max, res, seatdiff, drvn, drvln, guin, guiln, dep, ret};  
tbModel.addRow( rowData:tb_data );
```

## CheckOffers.java

### Κώδικας:

```
public void updateTable(){  
    try{  
        DefaultTableModel tbModel= (DefaultTableModel) CheckOffers.getModel();  
        Class.forName("com.mysql.cj.jdbc.Driver");  
        Connection con = DriverManager.getConnection("jdbc:mysql://localhost:3306/travel_agency?zeroDateTimeBehavior=CONVERT_TO_NULL", "root", "root");  
        String s = "{CALL check_offers('"+lname.getText()+"')}";  
        CallableStatement cs = con.prepareCall(s);  
        cs.execute();  
        ResultSet rs = cs.getResultSet();  
        tbModel.setRowCount(0);  
        while(rs.next()){  
            String id = rs.getString("Offer_ID");  
            String name = rs.getString("First_Name");  
            String ln = rs.getString("Last_Name");  
            String tb_data[] = {id, name, ln};  
            tbModel.addRow(tb_data);  
        }  
        if(tbModel.getRowCount()==0){JOptionPane.showMessageDialog(this, "There aren't any Reservations under this Last Name");}  
        con.close();  
    }catch(ClassNotFoundException | SQLException e){JOptionPane.showMessageDialog(this, e.getMessage());}  
}  
  
private void CancelActionPerformed(java.awt.event.ActionEvent evt) {  
    dispose();  
    Main main = new Main();  
    main.show();  
}  
  
private void SearchActionPerformed(java.awt.event.ActionEvent evt) {  
    if(!"".equals(lname.getText())){  
        updateTable();  
    }else JOptionPane.showMessageDialog(this, "Please Fill all the Fields in order to proceed!");  
}
```

## Επεξήγηση:

Αυτό το JFrame χρησιμοποιείται στην περίπτωση που ο IT Manager θέλει να προβάλει όλα τα reservation offers που αντιστοιχούν σε ένα συγκεκριμένο επώνυμο. Όπως και στα παραπάνω χρησιμοποιούνται συναρτήσεις όπως οι updateTable() και Cancel. Ωστόσο, σε αυτήν την περίπτωση η updateTable() είναι τροποποιημένη ώστε να καλεί το Stored Procedure check\_offers. Αναλυτικότερα, αρχικοποιούμε μία σύνδεση με την βάση δεδομένων, με ένα try-catch block και δημιουργούμε ένα CallableStatement το οποίο παίρνει σαν όρισμα το παρακάτω String:

```
String s = "{CALL check_offers('"+lname.getText()+"')}";
```

Τέλος, αποθηκεύουμε κάθε πεδίο κάθε εγγραφής του πίνακα σε μία μεταβλητή και προσθέτουμε μία γραμμή στο Model αυτού του jTable με αυτά τα πεδία, με τις παρακάτω εντολές:

```
String tb_data[] = {id, name, ln};  
tbModel.addRow( rowData:tb_data );
```

## Κώδικας:

```
static float cost = 0;
static int count = 0;
static int count2 = 0;
public void updateCombo(){
    try{
        Class.forName("com.mysql.cj.jdbc.Driver");
        Connection con = DriverManager.getConnection("jdbc:mysql://localhost:3306/travel_agency?zeroDateTimeBehavior=CONVERT_TO_NULL", "root", "root");
        String select="SELECT br_code FROM branch;";
        Statement slct = con.createStatement();
        ResultSet rs = slct.executeQuery(select);
        DefaultComboBoxModel mod = new DefaultComboBoxModel();
        mod.removeAllElements();
        while(rs.next()){
            String box = "Branch: "+rs.getInt("br_code")+"";
            mod.addElement(box);
        }
        br_code.setModel(mod);
        con.close();
    }catch(ClassNotFoundException | SQLException e){JOptionPane.showMessageDialog(this, e.getMessage());}
}

public void updateTable(){
    try{
        DefaultTableModel tbModel= (DefaultTableModel) BranchInfo.getModel();
        Class.forName("com.mysql.cj.jdbc.Driver");
        Connection con = DriverManager.getConnection("jdbc:mysql://localhost:3306/travel_agency?zeroDateTimeBehavior=CONVERT_TO_NULL", "root", "root");
        String code = br_code.getSelectedItem().toString().replaceAll("[^0-9.]", "");
        int i=Integer.parseInt(code);
        String select1="SELECT COUNT(tr_id), tr_cost FROM reservation INNER JOIN trip ON tr_id = res_tr_id WHERE tr_br_code=? GROUP BY tr_id;";
        PreparedStatement slct1 = con.prepareStatement(select1);
        slct1.setInt(1, i);
        ResultSet rs1 = slct1.executeQuery();
        String select2="SELECT br_street, br_num, br_city, wrk_name, wrk_lname FROM branch INNER JOIN worker ON br_code = wrk_br_code INNER JOIN admin ON wrk_AT = adm_AT INNER JOIN manages ON adm_AT = mng_adm_AT WHERE wrk_br_code="+i+"";
        PreparedStatement slct2 = con.prepareStatement(select2);
        ResultSet rs2 = slct2.executeQuery(select2);
        tbModel.setRowCount(0);
        while(rs1.next()){
            count = rs1.getInt("COUNT(tr_id)");
            count2 = count2 + rs1.getInt("COUNT(tr_id)");
            cost = cost + rs1.getFloat("tr_cost") * count;
        }
        while(rs2.next()){
            String street = rs2.getString("br_street");
            String num = rs2.getString("br_num");
            String city = rs2.getString("br_city");
            String name = rs2.getString("wrk_name");
            String ln = rs2.getString("wrk_lname");
            String costs = Float.toString(cost);
            String counts = Integer.toString(count2);
            String tb_data[] = {street, num, city, name, ln, counts, costs};
            tbModel.addRow(tb_data);
        }
        con.close();
    }catch(ClassNotFoundException | SQLException e){JOptionPane.showMessageDialog(this, e.getMessage());}
}

private void formWindowOpened(java.awt.event.WindowEvent evt) {
    updateCombo();
}

private void br_codeActionPerformed(java.awt.event.ActionEvent evt) {
    updateTable();
}

private void customButton1ActionPerformed(java.awt.event.ActionEvent evt) {
    dispose();
    Main main = new Main();
    main.show();
}
```

## Επεξήγηση:

Αυτό το JFrame χρησιμοποιείται στην περίπτωση που ο IT Manager θέλει να προβάλει τα στοιχεία ενός υποκαταστήματος, το ονοματεπώνυμο του διευθυντή του, το σύνολο κρατήσεων και το σύνολο εσόδων. Όπως και στα παραπάνω χρησιμοποιούνται συναρτήσεις όπως οι, updateCombo(), updateTable() και Cancel.

## Employees.java

### Κώδικας:

```
float total = 0;

public void updateCombo(){
    try{
        Class.forName("com.mysql.cj.jdbc.Driver");
        Connection con = DriverManager.getConnection("jdbc:mysql://localhost:3306/travel_agency?zeroDateTimeBehavior=CONVERT_TO_NULL", "root", "root");
        String select="SELECT br_code FROM branch;";
        Statement slct = con.createStatement();
        ResultSet rs = slct.executeQuery(select);
        DefaultComboBoxModel mod = new DefaultComboBoxModel();
        mod.removeAllElements();
        while(rs.next()){
            String box = "Branch: "+rs.getInt("br_code")+"";
            mod.addElement(box);
        }
        br_code.setModel(mod);
        con.close();
    }catch(ClassNotFoundException | SQLException e){JOptionPane.showMessageDialog(this, e.getMessage());}
}

public void updateTable(){
    String code = br_code.getSelectedItem().toString().replaceAll("[^0-9]", "");
    if(!"".equals(code)){
        try{
            DefaultTableModel tbModel= (DefaultTableModel) EmployeesTable.getModel();
            Class.forName("com.mysql.cj.jdbc.Driver");
            Connection con = DriverManager.getConnection("jdbc:mysql://localhost:3306/travel_agency?zeroDateTimeBehavior=CONVERT_TO_NULL", "root", "root");
            String select1="SELECT wrk_name, wrk_lname, wrk_salary FROM worker WHERE wrk_br_code = ?";
            int i=Integer.parseInt(code);
            PreparedStatement slct1 = con.prepareStatement(select1);
            slct1.setInt(1, i);
            ResultSet rs1 = slct1.executeQuery();
            tbModel.setRowCount(0);
            total = 0;
            while(rs1.next()){
                String name = rs1.getString("wrk_name");
                String lname = rs1.getString("wrk_lname");
                String salary = rs1.getString("wrk_salary");
                total = total + Float.parseFloat(salary);
                String tb_data[] = {name, lname, salary};
                tbModel.addRow(tb_data);
            }
            String str1 = String.format("%.02f", total);
            totalcost.setText(str1);
            if(tbModel.getRowCount()==0){JOptionPane.showMessageDialog(this, "There aren't any Employees");}
            con.close();
        }catch(ClassNotFoundException | SQLException e){JOptionPane.showMessageDialog(this, e.getMessage());}
    }
}

private void formWindowOpened(java.awt.event.WindowEvent evt) {
    updateCombo();
}

private void Cancel1ActionPerformed(java.awt.event.ActionEvent evt) {
    dispose();
    Main main = new Main();
    main.show();
}

private void br_codeActionPerformed(java.awt.event.ActionEvent evt) {
    updateTable();
}
```

## Επεξήγηση:

Αυτό το JFrame χρησιμοποιείται στην περίπτωση που ο IT Manager θέλει να προβάλει το όνομα, επώνυμο και μισθό όλων των υπάλληλων του και το συνολικό ποσό μισθών που πληρώνει. Όπως και στα παραπάνω χρησιμοποιούνται συναρτήσεις όπως οι, updateCombo(), updateTable() και Cancel.

[logs.java](#)

### Κώδικας:

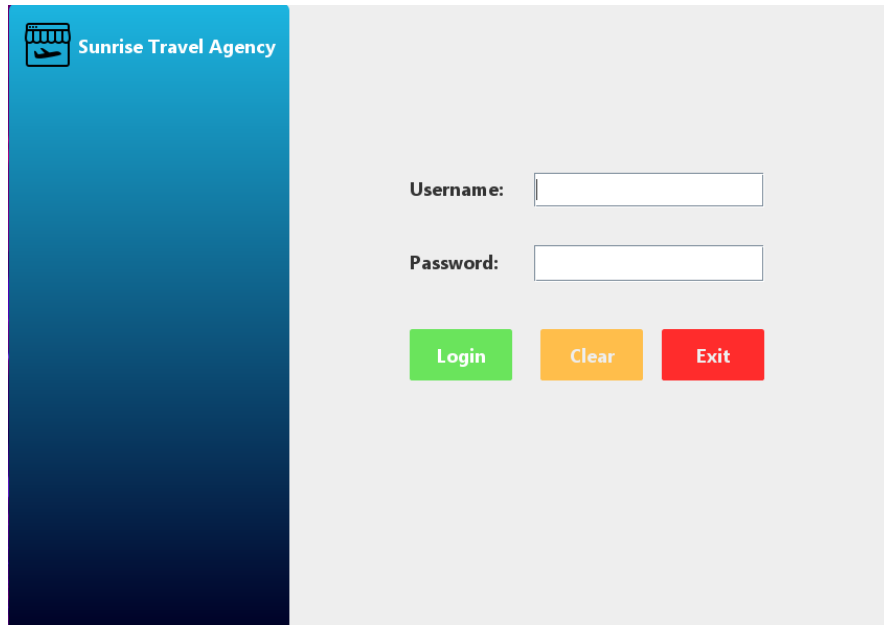
```
private void CancelActionPerformed(java.awt.event.ActionEvent evt) {  
    dispose();  
    Main main = new Main();  
    main.show();  
}  
public void updateTable(){  
    try{  
        DefaultTableModel tbModel= (DefaultTableModel) LogsTable.getModel();  
        Class.forName("com.mysql.cj.jdbc.Driver");  
        Connection con = DriverManager.getConnection("jdbc:mysql://localhost:3306/travel_agency?zeroDateTimeBehavior=CONVERT_TO_NULL", "root", "root");  
        String select="SELECT * FROM logging;";  
        Statement slct = con.createStatement();  
        ResultSet rs = slct.executeQuery(select);  
        tbModel.setRowCount(0);  
        while(rs.next()){  
            String change = rs.getString("chang");  
            String table = rs.getString("tab_ch");  
            String ln = rs.getString("it_lname");  
            String date = rs.getString("curr_date");  
            String tb_data[] = {change, table, ln, date};  
            tbModel.addRow(tb_data);  
        }  
        con.close();  
    }catch(ClassNotFoundException | SQLException e){JOptionPane.showMessageDialog(this, e.getMessage());}  
}
```

### Επεξήγηση:

Αυτό το JFrame χρησιμοποιείται στην περίπτωση που ο IT Manager θέλει να δει όλες τις ενέργειες που έχουν καταγραφεί στον πίνακα log: δηλαδή τον πίνακα στο οποίο έγινε κάποια ενέργεια, το είδος της ενέργειας, το επώνυμο του υπευθύνου Πληροφορικής που έκανε την ενέργεια και το Timestamp. Όπως και στα παραπάνω χρησιμοποιούνται συναρτήσεις όπως οι, updateTable() και Cancel.

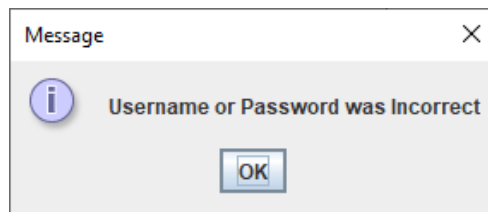
## Σενάριο Χρήσης

Μόλις ο χρήστης ανοίγει την εφαρμογή, βρίσκεται μπροστά σε ένα παράθυρο που του ζητάει να κάνει login με το username και το password του.



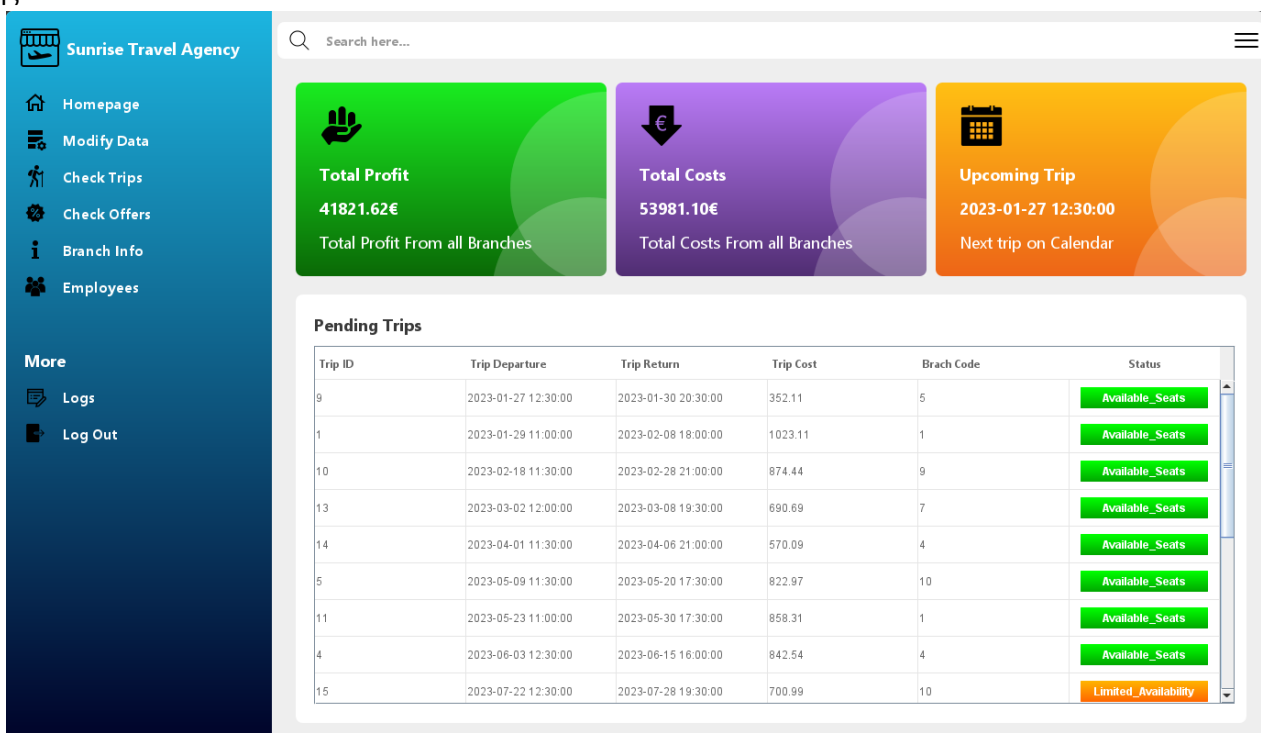
The login screen features a blue header with the 'Sunrise Travel Agency' logo. Below the header, there are two input fields: 'Username:' and 'Password:'. Underneath these fields are three buttons: 'Login' (green), 'Clear' (orange), and 'Exit' (red).

Εάν ο χρήστης εισάγει στοιχεία που δεν υπάρχουν ή είναι λανθασμένα και πατήσει το κουμπί login, τότε καθαρίζει τα πεδία username και password και εμφανίζεται το παρακάτω μήνυμα λάθους.



Εάν ο χρήστης πατήσει το κουμπί clear, τότε καθαρίζονται τα πεδία username και password και μπορεί να πληκτρολογήσει από την αρχή τα στοιχεία του, ενώ σε περίπτωση που ο χρήστης πατήσει το κουμπί Exit η εφαρμογή κλείνει.

Αν ο IT Manager εισάγει σωστά τα στοιχεία του, τότε εμφανίζεται μπροστά του ένα παράθυρο με το αρχικό menu της εφαρμογής.



The main dashboard of the application. On the left is a sidebar menu with the following items: Homepage, Modify Data, Check Trips, Check Offers, Branch Info, Employees, More, Logs, and Log Out. The main content area has a search bar at the top. Below it are three summary cards: 'Total Profit' (41821.62€), 'Total Costs' (53981.10€), and 'Upcoming Trip' (2023-01-27 12:30:00). Below these cards is a table titled 'Pending Trips' with columns: Trip ID, Trip Departure, Trip Return, Trip Cost, Brach Code, and Status. The table contains 15 rows of data.

Trip ID	Trip Departure	Trip Return	Trip Cost	Brach Code	Status
9	2023-01-27 12:30:00	2023-01-30 20:30:00	352.11	5	Available_Seats
1	2023-01-29 11:00:00	2023-02-08 18:00:00	1023.11	1	Available_Seats
10	2023-02-18 11:30:00	2023-02-28 21:00:00	874.44	9	Available_Seats
13	2023-03-02 12:00:00	2023-03-08 19:30:00	690.69	7	Available_Seats
14	2023-04-01 11:30:00	2023-04-06 21:00:00	570.09	4	Available_Seats
5	2023-05-09 11:30:00	2023-05-20 17:30:00	822.97	10	Available_Seats
11	2023-05-23 11:00:00	2023-05-30 17:30:00	858.31	1	Available_Seats
4	2023-06-03 12:30:00	2023-06-15 16:00:00	842.54	4	Available_Seats
15	2023-07-22 12:30:00	2023-07-28 19:30:00	700.99	10	Limited_Availability

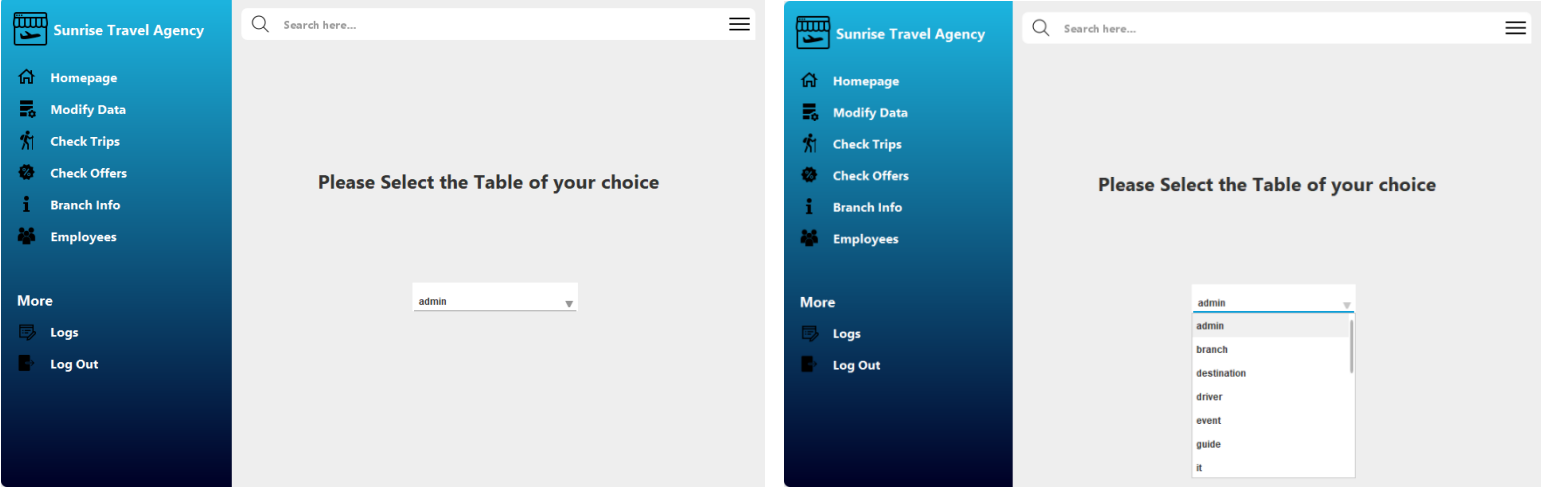
Σε αυτό το παράθυρο, βλέπουμε αριστερά μία λίστα με τις δυνατότητες που μπορεί να προσφέρει η εφαρμογή στον IT Manager, ενώ δεξιά βλέπουμε στο πάνω μέρος 3 καρτέλες με έξτρα πληροφορίες όπως το Συνολικό Κέρδος του πρακτορείου,



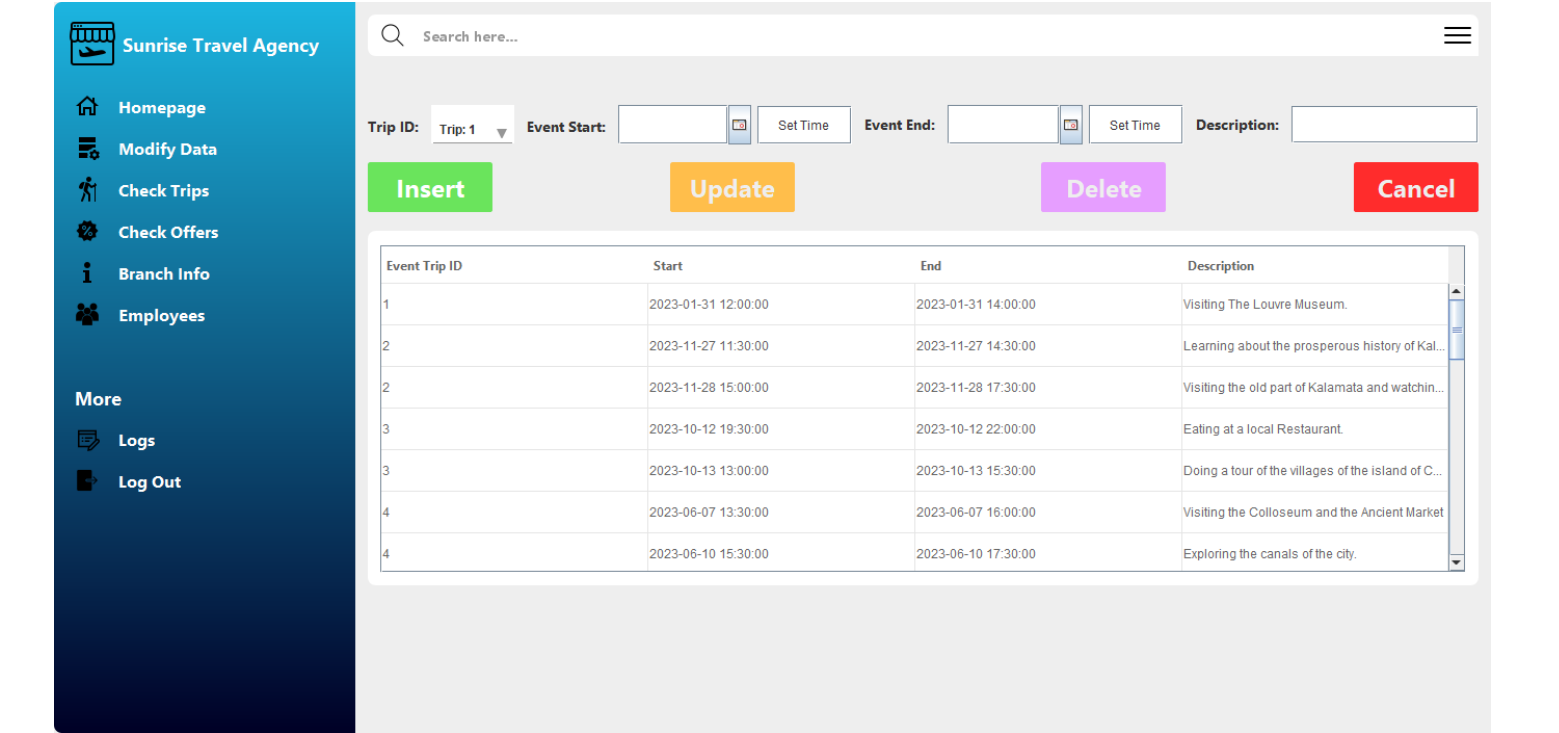
τα Συνολικά Έξοδα του πρακτορείου και το Επόμενο Ταξίδι του πρακτορείου. Στο κάτω μέρος της δεξιάς πλευράς, παρατηρούμε πως υπάρχει ένας πίνακας στον οποίο φαίνονται όλα τα ταξίδια του πρακτορείου. Για το κάθε ταξίδι εμφανίζεται το id του, οι ημερομηνίες αναχώρησης και επιστροφής, το κόστος ανά άτομο, το υποκατάστημα το οποίο το διοργανώνει, καθώς και ένα label που δείχνει την διαθεσιμότητα των θέσεων του ταξιδιού.

Ο IT Manager τώρα, μπορεί να πατήσει ένα από τα κουμπιά στα αριστερά του παραθύρου και να κάνει μία από τις αντίστοιχες ενέργειες.

Πατώντας το κουμπί Modify Data, ο IT Manager βλέπει το παρακάτω παράθυρο.

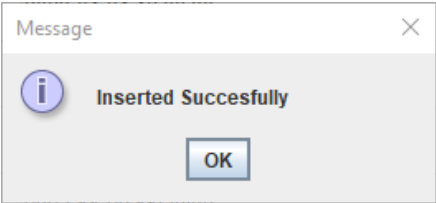


Ο IT Manager τώρα μπορεί να διαλέξει τον πίνακα που επιθυμεί να επεξεργαστεί, με την βοήθεια ενός Drop Down Menu. Για το παράδειγμα μας, ας υποθέσουμε ότι ο χρήστης θέλει να επεξεργαστεί τον πίνακα event. Τότε θα εμφανιστεί μπροστά του το παρακάτω παράθυρο.



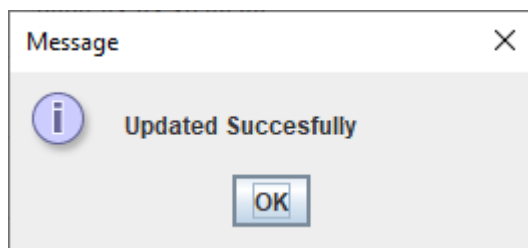
Τώρα έχει την δυνατότητα να κάνει 4 ενέργειες σε αυτήν την σελίδα.

- **Insert:** Για να κάνει Insert στον πίνακα που διάλεξε στο προηγούμενο βήμα, συμπληρώνει τα πεδία που υπάρχουν στο πάνω μέρος της σελίδας και έπειτα πατάει το κουμπί Insert. Αν η εισαγωγή ήταν επιτυχής τότε εμφανίζεται ένα μήνυμα στην οθόνη και εμφανίζεται η νέα εγγραφή στον πίνακα.

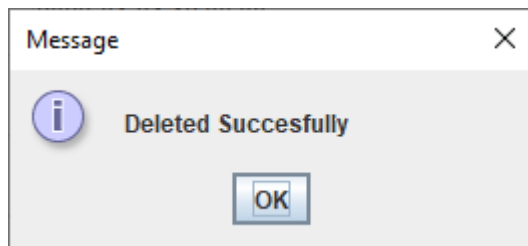




- **Update:** Για να κάνει Update σε μία εγγραφή του πίνακα, ο IT Manager πρέπει να την επιλέξει από τον πίνακα πατώντας πάνω της. Όταν το κάνει, τα παραπάνω πεδία γεμίζουν με τις τιμές της επιλεγμένης εγγραφής και ο IT Manager μπορεί να τα επεξεργαστεί. Όταν είναι ικανοποιημένος με τις αλλαγές που έκανε, πατάει το κουμπί Update και ο πίνακας ενημερώνεται. Επίσης εμφανίζεται και μήνυμα που τον επιβεβαιώνει ότι η ενημέρωση ολοκληρώθηκε με επιτυχία.



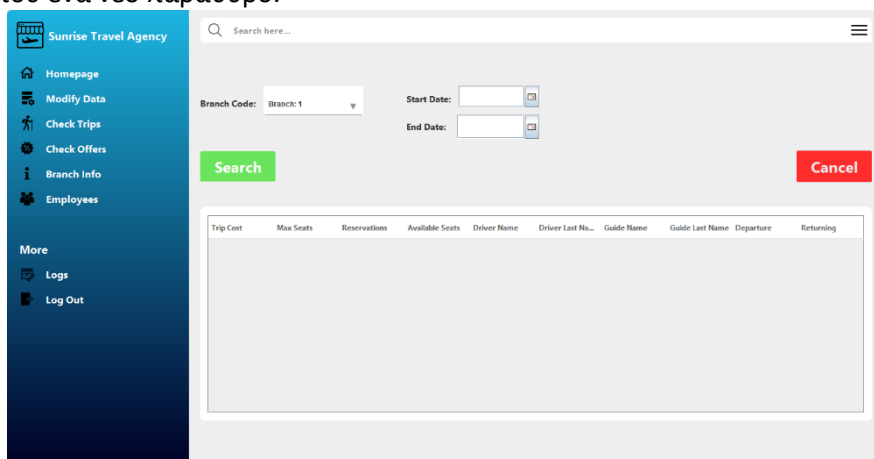
- **Delete:** Για να κάνει Delete μια εγγραφή του πίνακα, ο IT Manager πρέπει να την επιλέξει από τον πίνακα πατώντας πάνω της. Έπειτα το μόνο που έχει να κάνει είναι να πατήσει το κουμπί Delete και η εγγραφή θα διαγραφεί από τον πίνακα. Επίσης θα εμφανιστεί στην οθόνη του και κατάλληλο μήνυμα το οποίο τον επιβεβαιώνει για την επιτυχία της διαγραφής του.



- **Cancel:** Αν ο χρήστης πατήσει το κουμπί Cancel, τότε κλείνει αυτό το παράθυρο και ο χρήστης επιστρέφει στο αρχικό menu της εφαρμογής.

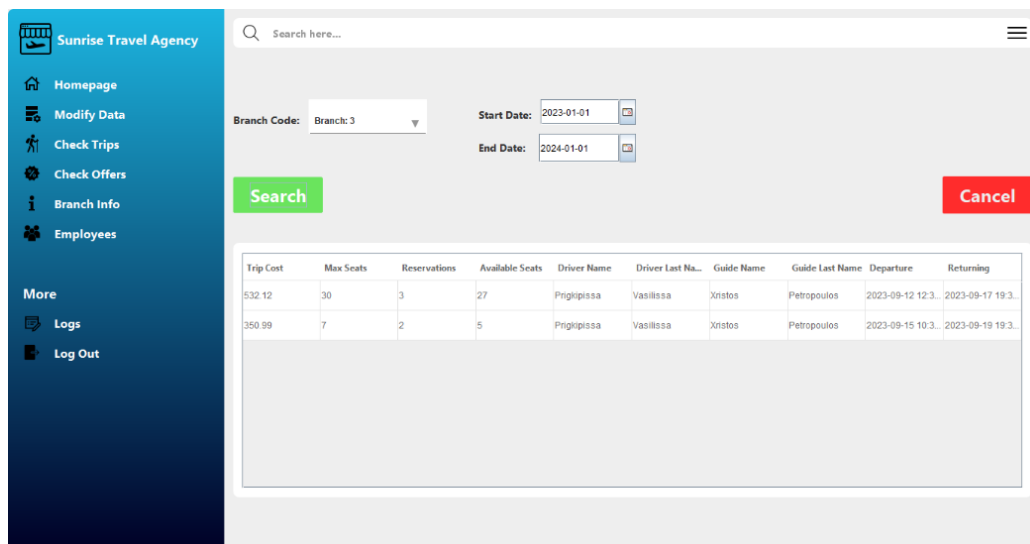
**Έπειτα ο IT Manager, επιλέγει από τα αριστερά το κουμπί Check Trips.**

**Θα εμφανιστεί μπροστά του ένα νέο παράθυρο.**




Σε αυτό το παράθυρο, ο IT Manager μπορεί να αναζητήσει με βάση το Branch και τις ημερομηνίες αναχώρησης και επιστροφής, τα διαθέσιμα ταξίδια. Αυτό το πραγματοποιεί συμπληρώνοντας τα πεδία στο πάνω μέρος της σελίδας και πατώντας το κουμπί Search.

Στην συνέχεια πατώντας το κουμπί Cancel, μπορεί να επιστρέψει στο αρχικό Menu.



Στην συνέχεια ο IT Manager επιλέγει από τα αριστερά το κουμπί Check Offers.

Εμφανίζεται μπροστά του το παρακάτω παράθυρο.

Sunrise Travel Agency

Homepage

Modify Data

Check Trips

Check Offers

Branch Info

Employees

More

Logs

Log Out

Search here...

Last Name:


Search

Cancel

Offer ID	First Name	Last Name
----------	------------	-----------

Σε αυτό το παράθυρο ο χρήστης έχει την δυνατότητα, πληκτρολογώντας ένα επώνυμο, να δει ποια reservation offers έχουν γίνει με αυτό το επώνυμο. Το μόνο που έχει να κάνει είναι να πληκτρολογήσει το επώνυμο στο πεδίο στο πάνω μέρος της σελίδας και να πατήσει το κουμπί Search.

Στην συνέχεια πατώντας το κουμπί Cancel, μπορεί να επιστρέψει στο αρχικό Menu.

Sunrise Travel Agency

Homepage

Modify Data

Check Trips

Check Offers

Branch Info

Employees

More

Logs

Log Out

Search here...

Last Name:


Search

Cancel

Offer ID	First Name	Last Name
1	Monique	Baldwin
1	Kylee	Baldwin
1	Kiera	Baldwin
1	Jocelyn	Baldwin
1	Amari	Baldwin
1	Kaleb	Baldwin
1	Jamya	Baldwin

Στην συνέχεια ο IT Manager επιλέγει από τα αριστερά το κουμπί Branch Info.

Εμφανίζεται μπροστά του το παρακάτω παράθυρο.

Sunrise Travel Agency

Homepage

Modify Data

Check Trips

Check Offers

Branch Info

Employees

More

Logs

Log Out

Search here...


Branch Code: 

Branch: 1

Cancel

Street	Number	City	Name	Last Name	Reservations	Revenue
--------	--------	------	------	-----------	--------------	---------

Εδώ ο IT Manager έχει την επιλογή να διαλέξει ένα Branch, με βάση το Branch Code του. Όταν κάνει την επιλογή του, στον παρακάτω πίνακα θα εμφανιστούν οι πληροφορίες που αντιστοιχούν στο συγκεκριμένο Branch.

 Sunrise Travel Agency

Homepage

Modify Data

Check Trips

Check Offers

Branch Info

Employees

More

Logs

Log Out


Search here...

Branch Code: Branch: 4

Cancel

Street	Number	City	Name	Last Name	Reservations	Revenue
Zaimi	30	Patra	Davon	Sampson	4	2825.26

Στην συνέχεια ο IT Manager επιλέγει από τα αριστερά το κουμπί Employees. Εμφανίζεται μπροστά του το παρακάτω παράθυρο.

 Sunrise Travel Agency

Homepage

Modify Data

Check Trips

Check Offers

Branch Info

Employees

More

Logs

Log Out

Search here...


Branch Code: Branch: 1

Total Costs:

Cancel

Employer Name	Employer Last Name	Salary
---------------	--------------------	--------

Με την βοήθεια αυτού του παραθύρου, ο IT Manager, έχει την δυνατότητα να δει όλους τους υπαλλήλους που υπάρχουν σε ένα Branch, καθώς και το σύνολο των μισθών τους. Το μόνο που έχει να κάνει για να το επιτύχει είναι να διαλέξει από το Drop Down Menu, το υποκατάστημα της αρέσκειας του.

 Sunrise Travel Agency

Homepage

Modify Data

Check Trips

Check Offers

Branch Info

Employees

More

Logs

Log Out

Search here...


Branch Code: Branch: 3

Total Costs: 3795.92

Cancel

Employer Name	Employer Last Name	Salary
Andreas	Olkonomou	1233.2
Prigkipissa	Vasilissa	655.2
Efi	Persikou	942.3
Xristos	Petropoulos	965.22

Στην συνέχεια ο IT Manager επιλέγει από τα αριστερά το κουμπί Logs.  
Εμφανίζεται μπροστά του το παρακάτω παράθυρο.

Sunrise Travel Agency

Homepage

Modify Data

Check Trips

Check Offers

Branch Info

Employees

More

Logs

Log Out

Search here...

Cancel

Action	Table	IT Last Name	Timestamp
INSERT	event	Archer	2023-01-10 16:24:09
UPDATE	event	Archer	2023-01-10 16:24:28
DELETE	event	Archer	2023-01-10 16:24:32

Μπροστά του τότε εμφανίζονται όλες οι κινήσεις που έκανε ο ίδιος κατά την χρήση της εφαρμογής, καθώς και οι κινήσεις που είχαν κάνει πριν από αυτόν (αν υπάρχουν).  
Τέλος, πατώντας το κουμπί Log out, επιστρέφει στην αρχική σελίδα στην οποία μπορεί να κάνει login και αν επιθυμεί κλείνει το πρόγραμμα πατώντας το κουμπί Exit.

## Κεφάλαιο 5

### InsertSelection2.java

#### Κώδικας:

```
private void Selection1ActionPerformed(java.awt.event.ActionEvent evt) {
    String select = Selection1.getSelectedItemId().toString();
    switch(select){
        case "Worker":
            dispose();
            wrk_mng inspage3 = new wrk_mng();
            inspage3.show();
            break;
        case "Driver":
            dispose();
            drv_mng inspage5 = new drv_mng();
            inspage5.show();
            break;
        case "Admin":
            dispose();
            adm_mng inspage6 = new adm_mng();
            inspage6.show();
            break;
        case "Guide":
            dispose();
            gui_mng inspage8 = new gui_mng();
            inspage8.show();
            break;
        case "Trip":
            dispose();
            trip_mng inspage10 = new trip_mng();
            inspage10.show();
            break;
    }
}
```

### Employees\_mng.java

#### Κώδικας:

```
float total = 0;
public void updateTable(){
    try{
        DefaultTableModel tbModel= (DefaultTableModel) EmployeesTable.getModel();
        Class.forName("com.mysql.cj.jdbc.Driver");
        Connection con = DriverManager.getConnection("jdbc:mysql://localhost:3306/travel_agency?zeroDateTimeBehavior=CONVERT_TO_NULL", "root",
"root");
        String select1="SELECT wrk_name, wrk_lname, wrk_salary FROM worker WHERE wrk_br_code = "+Login.getBranch()+"";
        PreparedStatement slct1 = con.prepareStatement(select1);
        ResultSet rs1 = slct1.executeQuery();
        tbModel.setRowCount(0);
        total = 0;
        while(rs1.next()){
            String name = rs1.getString("wrk_name");
            String lname = rs1.getString("wrk_lname");
            String salary = rs1.getString("wrk_salary");
            total = total + Float.parseFloat(salary);
            String tb_data[] = {name, lname, salary};
            tbModel.addRow(tb_data);
        }
        String str1 = String.format("%.02f", total);
        totalcost.setText(str1);
        if(tbModel.getRowCount()==0){JOptionPane.showMessageDialog(this, "There aren't any Employees");}
        con.close();
    }catch(ClassNotFoundException | SQLException e){JOptionPane.showMessageDialog(this, e.getMessage());}
}

private void formWindowOpened(java.awt.event.WindowEvent evt) {
    br_code.setText(Login.getBranch());
    updateTable();
}

private void Cancel1ActionPerformed(java.awt.event.ActionEvent evt) {
    dispose();
    Main main = new Main();
    main.show();
}
```

**Κώδικας:**

```
public void updateTable(){
    try{
        Class.forName("com.mysql.cj.jdbc.Driver");
        Connection con = DriverManager.getConnection("jdbc:mysql://localhost:3306/travel_agency?zeroDateTimeBehavior=CONVERT_TO_NULL", "root", "root");
        String select="SELECT * FROM admin;";
        Statement slct = con.createStatement();
        ResultSet rs = slct.executeQuery(select);
        DefaultTableModel tbModel= (DefaultTableModel) AdminTable.getModel();
        tbModel.setNumRows(0);
        while(rs.next()){
            String at = rs.getString("adm_AT");
            String language = rs.getString("adm_type");
            String diploma = rs.getString("adm_diploma");
            String tb_data[] = {at,language,diploma};
            tbModel.addRow(tb_data);
        }
        con.close();
    }catch(ClassNotFoundException | SQLException e){JOptionPane.showMessageDialog(this, e.getMessage());}
}

public void updateCombo(){
    try{
        Class.forName("com.mysql.cj.jdbc.Driver");
        Connection con = DriverManager.getConnection("jdbc:mysql://localhost:3306/travel_agency?zeroDateTimeBehavior=CONVERT_TO_NULL", "root", "root");
        String select ="SELECT wrk_AT FROM worker;";
        Statement slct = con.createStatement();
        ResultSet rs = slct.executeQuery(select);
        DefaultComboBoxModel mod = new DefaultComboBoxModel();
        mod.removeAllElements();
        while(rs.next()){mod.addElement(rs.getString("wrk_AT"));}
        wrk_AT.setModel(mod);
        con.close();
    }catch(ClassNotFoundException | SQLException e){JOptionPane.showMessageDialog(this, e.getMessage());}
}

private void formWindowOpened(java.awt.event.WindowEvent evt) {
    updateCombo();
    updateTable();
}

private void AdminTableMouseClicked(java.awt.event.MouseEvent evt) {
    DefaultTableModel tbModel= (DefaultTableModel) AdminTable.getModel();
    String at = tbModel.getValueAt(AdminTable.getSelectedRow(), 0).toString();
    String type = tbModel.getValueAt(AdminTable.getSelectedRow(), 1).toString();
    String diploma = tbModel.getValueAt(AdminTable.getSelectedRow(), 2).toString();
    wrk_AT.setSelectedItem(at);
    adm_type.setSelectedItem(type);
    adm_diploma.setText(diploma);
}

private void InsertActionPerformed(java.awt.event.ActionEvent evt) {
    try{
        Class.forName("com.mysql.cj.jdbc.Driver");
        Connection con = DriverManager.getConnection("jdbc:mysql://localhost:3306/travel_agency?zeroDateTimeBehavior=CONVERT_TO_NULL", "root", "root");
        String insert="INSERT INTO admin(adm_AT,adm_diploma,adm_type) VALUES(?,?,?)";
        PreparedStatement insrt = con.prepareStatement(insert);
        insrt.setString(1,wrk_AT.getSelectedItem().toString());
        insrt.setString(2,adm_diploma.getText());
        insrt.setString(3,adm_type.getSelectedItem().toString());
        insrt.execute();
        updateTable();
        JOptionPane.showMessageDialog(this, "Inserted Succesfully");
    }catch(ClassNotFoundException | SQLException e){JOptionPane.showMessageDialog(this, e.getMessage());}
}

private void CancelActionPerformed(java.awt.event.ActionEvent evt) {
    dispose();
    Main main = new Main();
    main.show();
}

private void DeleteActionPerformed(java.awt.event.ActionEvent evt) {
    try{
        String fname = null;
        String lname = null;
        String test = null;
        String msg = "You cannot Delete this User. This User is an Administrative.";
```

```

String msg2 = "This Worker is not an Admin!";
Class.forName("com.mysql.cj.jdbc.Driver");
Connection con = DriverManager.getConnection("jdbc:mysql://localhost:3306/travel_agency?zeroDateTimeBehavior=CONVERT_TO_NULL", "root", "root");
String delete="DELETE FROM admin WHERE adm_AT = ?";
String select1="SELECT wrk_name FROM worker INNER JOIN admin ON adm_AT=wrk_AT WHERE wrk_AT = ? AND adm_type='ADMINISTRATIVE'";
String select2="SELECT wrk_lname FROM worker INNER JOIN admin ON adm_AT=wrk_AT WHERE wrk_AT = ? AND adm_type='ADMINISTRATIVE'";
String select3="SELECT adm_AT FROM admin";
PreparedStatement dlt = con.prepareStatement(delete);
PreparedStatement slc1 = con.prepareStatement(select1);
PreparedStatement slc2 = con.prepareStatement(select2);
PreparedStatement slc3 = con.prepareStatement(select3);
slc1.setString(1, wrk_AT.getSelectedItem().toString());
slc2.setString(1, wrk_AT.getSelectedItem().toString());
dlt.setString(1, wrk_AT.getSelectedItem().toString());
ResultSet rs1 = slc1.executeQuery();
ResultSet rs2 = slc2.executeQuery();
ResultSet rs4 = slc3.executeQuery();
if(rs1.next()==true){fname = rs1.getString("wrk_name");}
if(rs2.next()==true){lname = rs2.getString("wrk_lname");}
String s = "{CALL admin_check('"+fname+"','"+lname+"')}";
CallableStatement cs = con.prepareCall(s);
cs.execute();
ResultSet rs3 = cs.getResultSet();
System.out.println(fname);
System.out.println(lname);
if(rs3 != null)rs3.next();
if(rs3.getString(1).equals(msg)){
    JOptionPane.showMessageDialog(this, msg);
}else if(rs3.getString(1).equals(msg2) && fname == null){
    while(rs4.next()){
        if(rs4.getString("adm_AT").equals(wrk_AT.getSelectedItem().toString())){
            test = "found";
            break;
        }else test = "not found";
    }if ("found".equals(test)){
        dlt.execute();
        JOptionPane.showMessageDialog(this, "Deleted Succesfully");
    }else JOptionPane.showMessageDialog(this, msg2);
    }
    updateTable();
    con.close();
}catch(ClassNotFoundException | SQLException e){JOptionPane.showMessageDialog(this, e.getMessage());}
}

private void UpdateActionPerformed(java.awt.event.ActionEvent evt) {
try{
    DefaultTableModel tbModel= (DefaultTableModel) AdminTable.getModel();
    Class.forName("com.mysql.cj.jdbc.Driver");
    Connection con = DriverManager.getConnection("jdbc:mysql://localhost:3306/travel_agency?zeroDateTimeBehavior=CONVERT_TO_NULL", "root", "root");
    String update="UPDATE admin SET adm_type = ?, adm_diploma = ? WHERE adm_AT = ?";
    PreparedStatement upd = con.prepareStatement(update);
    upd.setString(1, adm_type.getSelectedItem().toString());
    upd.setString(2, adm_diploma.getText());
    upd.setString(3, tbModel.getValueAt(AdminTable.getSelectedRow(), 0).toString());
    upd.executeUpdate();
    updateTable();
    JOptionPane.showMessageDialog(this, "Updated Succesfully");
    con.close();
}catch(ClassNotFoundException | SQLException e){JOptionPane.showMessageDialog(this, e.getMessage());}
}
}

```

## [drv\\_mng.java](#)

### Κώδικας:

```

public void updateTable(){
try{
    Class.forName("com.mysql.cj.jdbc.Driver");
    Connection con = DriverManager.getConnection("jdbc:mysql://localhost:3306/travel_agency?zeroDateTimeBehavior=CONVERT_TO_NULL", "root", "root");
    String select="SELECT * FROM driver";
    Statement slct = con.createStatement();
    ResultSet rs = slct.executeQuery(select);
    DefaultTableModel tbModel= (DefaultTableModel) DriverTable.getModel();
    tbModel.setNumRows(0);
    while(rs.next()){
        String at = rs.getString("drv_AT");
        String license = rs.getString("drv_license");
    }
}
}

```

```

        String route = rs.getString("drv_route");
        String experience = rs.getString("drv_experience");
        String tb_data[] = {at, license, route, experience};
        tbModel.addRow(tb_data);
    }
    con.close();
} catch (ClassNotFoundException | SQLException e) { JOptionPane.showMessageDialog(this, e.getMessage()); }
}

public void updateCombo(){
    try{
        Class.forName("com.mysql.cj.jdbc.Driver");
        Connection con = DriverManager.getConnection("jdbc:mysql://localhost:3306/travel_agency?zeroDateTimeBehavior=CONVERT_TO_NULL", "root", "root");
        String select = "SELECT wrk_AT FROM worker;";
        Statement slct = con.createStatement();
        ResultSet rs = slct.executeQuery(select);
        DefaultComboBoxModel mod = new DefaultComboBoxModel();
        mod.removeAllElements();
        while(rs.next()){mod.addElement(rs.getString("wrk_AT"));}
        wrk_AT.setModel(mod);
        con.close();
    } catch (ClassNotFoundException | SQLException e) { JOptionPane.showMessageDialog(this, e.getMessage()); }
}

private void formWindowOpened(java.awt.event.WindowEvent evt) {
    updateCombo();
    updateTable();
}

private void DriverTableMouseClicked(java.awt.event.MouseEvent evt) {
    DefaultTableModel tbModel = (DefaultTableModel) DriverTable.getModel();
    String at = tbModel.getValueAt(DriverTable.getSelectedRow(), 0).toString();
    String license = tbModel.getValueAt(DriverTable.getSelectedRow(), 1).toString();
    String route = tbModel.getValueAt(DriverTable.getSelectedRow(), 2).toString();
    String exp = tbModel.getValueAt(DriverTable.getSelectedRow(), 3).toString();
    wrk_AT.setSelectedItem(at);
    drv_license.setSelectedItem(license);
    drv_route.setSelectedItem(route);
    drv_experience.setText(exp);
}

private void InsertActionPerformed(java.awt.event.ActionEvent evt) {
    try{
        Class.forName("com.mysql.cj.jdbc.Driver");
        Connection con = DriverManager.getConnection("jdbc:mysql://localhost:3306/travel_agency?zeroDateTimeBehavior=CONVERT_TO_NULL", "root", "root");
        String insert = "INSERT INTO driver(drv_AT,drv_license,drv_route,drv_experience) VALUES(?,?,?,?)";
        PreparedStatement insrt = con.prepareStatement(insert);
        insrt.setString(1, wrk_AT.getSelectedItem().toString());
        insrt.setString(2, drv_license.getSelectedItem().toString());
        insrt.setString(3, drv_route.getSelectedItem().toString());
        insrt.setInt(4, Integer.parseInt(drv_experience.getText()));
        insrt.execute();
        updateTable();
        JOptionPane.showMessageDialog(this, "Inserted Succesfully");
    } catch (ClassNotFoundException | SQLException e) { JOptionPane.showMessageDialog(this, e.getMessage()); }
}

private void CancelActionPerformed(java.awt.event.ActionEvent evt) {
    dispose();
    Main main = new Main();
    main.show();
}

private void DeleteActionPerformed(java.awt.event.ActionEvent evt) {
    try{
        Class.forName("com.mysql.cj.jdbc.Driver");
        Connection con = DriverManager.getConnection("jdbc:mysql://localhost:3306/travel_agency?zeroDateTimeBehavior=CONVERT_TO_NULL", "root", "root");
        String delete = "DELETE FROM driver WHERE drv_AT = ?";
        PreparedStatement dlt = con.prepareStatement(delete);
        dlt.setString(1, wrk_AT.getSelectedItem().toString());
        dlt.executeUpdate();
        updateTable();
        wrk_AT.setSelectedItem("");
        drv_experience.setText("");
        JOptionPane.showMessageDialog(this, "Deleted Succesfully");
        con.close();
    } catch (ClassNotFoundException | SQLException e) { JOptionPane.showMessageDialog(this, e.getMessage()); }
}

```



```

private void UpdateActionPerformed(java.awt.event.ActionEvent evt) {
    try{
        Class.forName("com.mysql.cj.jdbc.Driver");
        Connection con = DriverManager.getConnection("jdbc:mysql://localhost:3306/travel_agency?zeroDateTimeBehavior=CONVERT_TO_NULL", "root", "root");
        String update="UPDATE driver SET drv_license = ?, drv_route = ?, drv_experience = ? WHERE drv_AT = ?";
        PreparedStatement upd = con.prepareStatement(update);
        upd.setString(1, drv_license.getSelectedItem().toString());
        upd.setString(2, drv_route.getSelectedItem().toString());
        upd.setInt(3, Integer.parseInt(drv_experience.getText()));
        upd.setString(4, wrk_AT.getSelectedItem().toString());
        upd.executeUpdate();
        updateTable();
        JOptionPane.showMessageDialog(this, "Updated Succesfully");
        con.close();
    }catch(ClassNotFoundException | SQLException e){JOptionPane.showMessageDialog(this, e.getMessage());}
}

```

## gui\_mng.java

### Κώδικας:

```

public void updateTable(){
    try{
        Class.forName("com.mysql.cj.jdbc.Driver");
        Connection con = DriverManager.getConnection("jdbc:mysql://localhost:3306/travel_agency?zeroDateTimeBehavior=CONVERT_TO_NULL", "root", "root");
        String select2="SELECT * FROM guide;";
        Statement slct2 = con.createStatement();
        ResultSet rs2 = slct2.executeQuery(select2);
        DefaultTableModel tbModel= (DefaultTableModel) GuiTable.getModel();
        tbModel.setNumRows(0);
        while(rs2.next()){
            String at = rs2.getString("gui_AT");
            String cv = rs2.getString("gui_cv");
            String tb_data[] = {at,cv};
            tbModel.addRow(tb_data);
        }
        con.close();
    }catch(ClassNotFoundException | SQLException e){JOptionPane.showMessageDialog(this, e.getMessage());}
}

public void updateCombo(){
    try{
        Class.forName("com.mysql.cj.jdbc.Driver");
        Connection con = DriverManager.getConnection("jdbc:mysql://localhost:3306/travel_agency?zeroDateTimeBehavior=CONVERT_TO_NULL", "root", "root");
        String select="SELECT wrk_AT FROM worker;";
        Statement slct = con.createStatement();
        ResultSet rs = slct.executeQuery(select);
        DefaultComboBoxModel mod = new DefaultComboBoxModel();
        mod.removeAllElements();
        while(rs.next()){mod.addElement(rs.getString("wrk_AT"));}
        updateTable();
        gui_AT.setModel(mod);
        con.close();
    }catch(ClassNotFoundException | SQLException e){JOptionPane.showMessageDialog(this, e.getMessage());}
}

private void formWindowOpened(java.awt.event.WindowEvent evt) {
    updateCombo();
}

private void GuiTableMouseClicked(java.awt.event.MouseEvent evt) {
    DefaultTableModel tbModel= (DefaultTableModel) GuiTable.getModel();
    String gui_at = tbModel.getValueAt(GuiTable.getSelectedRow(), 0).toString();
    String gui_CV = tbModel.getValueAt(GuiTable.getSelectedRow(), 1).toString();
    gui_AT.setSelectedItem(gui_at);
    gui_cv.setText(gui_CV);
}

private void InsertActionPerformed(java.awt.event.ActionEvent evt) {
    try{
        Class.forName("com.mysql.cj.jdbc.Driver");
        Connection con = DriverManager.getConnection("jdbc:mysql://localhost:3306/travel_agency?zeroDateTimeBehavior=CONVERT_TO_NULL", "root", "root");
        String insert="INSERT INTO guide(gui_AT,gui_cv) VALUES(?,?)";
        PreparedStatement insrt = con.prepareStatement(insert);
        insrt.setString(1,gui_AT.getSelectedItem().toString());
        insrt.setString(2,gui_cv.getText());
        insrt.execute();
        updateTable();
    }
}

```

```

        JOptionPane.showMessageDialog(this, "Inserted Successfully");
    }catch(ClassNotFoundException | SQLException e){JOptionPane.showMessageDialog(this, e.getMessage());}
}

private void CancelActionPerformed(java.awt.event.ActionEvent evt) {
    dispose();
    Main main = new Main();
    main.show();
}

private void Delete1ActionPerformed(java.awt.event.ActionEvent evt) {
    try{
        Class.forName("com.mysql.cj.jdbc.Driver");
        Connection con = DriverManager.getConnection("jdbc:mysql://localhost:3306/travel_agency?zeroDateTimeBehavior=CONVERT_TO_NULL", "root", "root");
        String delete="DELETE FROM guide WHERE gui_AT = ?";
        PreparedStatement dlt = con.prepareStatement(delete);
        dlt.setString(1, gui_AT.getSelectedItemAt().toString());
        dlt.executeUpdate();
        updateTable();
        gui_cv.setText("");
        JOptionPane.showMessageDialog(this, "Deleted Successfully");
        con.close();
    }catch(ClassNotFoundException | SQLException e){JOptionPane.showMessageDialog(this, e.getMessage());}
}

private void Update1ActionPerformed(java.awt.event.ActionEvent evt) {
    try{
        Class.forName("com.mysql.cj.jdbc.Driver");
        Connection con = DriverManager.getConnection("jdbc:mysql://localhost:3306/travel_agency?zeroDateTimeBehavior=CONVERT_TO_NULL", "root", "root");
        String update="UPDATE guide SET gui_cv = ? WHERE gui_AT = ?";
        PreparedStatement upd = con.prepareStatement(update);
        upd.setString(1, gui_cv.getText());
        upd.setString(2, gui_AT.getSelectedItemAt().toString());
        upd.executeUpdate();
        updateTable();
        JOptionPane.showMessageDialog(this, "Updated Successfully");
        con.close();
    }catch(ClassNotFoundException | SQLException e){JOptionPane.showMessageDialog(this, e.getMessage());}
}
}

```

[trip\\_mng.java](#)

**Κώδικας:**

```

static Timestamp t1;
static Timestamp t2;
static Timestamp temp;
public void convertTimeStamp(){
    String time1 = PickTime1.getText();
    String time2 = PickTime2.getText();
    time1=time1.replace(" ", "");
    time2=time2.replace(" ", "");
    if(time1.contains("PM")){
        String[] parts = time1.split(":");
        String part1 = parts[0];
        String part2 = parts[1];
        if(!"12".equals(part1)){
            int part1int = Integer.parseInt(part1)+12;
            part1 = Integer.toString(part1int);
            time1 = part1.concat(":").concat(part2);
        }else time1 = part1.concat(":").concat(part2);
    }else{
        String[] parts = time1.split(":");
        String part1 = parts[0];
        String part2 = parts[1];
        if("12".equals(part1)){
            part1 = "00";
            time1 = part1.concat(":").concat(part2);
        }else time1 = part1.concat(":").concat(part2);
    }
    time1=time1.replace("PM", "");
    time1=time1.replace("AM", "");

    if(time2.contains("PM")){
        String[] parts = time2.split(":");
        String part1 = parts[0];

```

```

String part2 = parts[1];
if(!"12".equals(part1)){
int part1int = Integer.parseInt(part1)+12;
part1 = Integer.toString(part1int);
time2 = part1.concat(":").concat(part2);
}else time2 = part1.concat(":").concat(part2);
}else{
String[] parts = time2.split(":");
String part1 = parts[0];
String part2 = parts[1];
if("12".equals(part1)){
part1 = "00";
time2 = part1.concat(":").concat(part2);
}else time2 = part1.concat(":").concat(part2);
}
time2=time2.replace("PM", "");
time2=time2.replace("AM", "");
time1=time1.concat(":00");
time2=time2.concat(":00");
DateFormat dateFormat1 = new SimpleDateFormat("yyyy-MM-dd ");
String strDate1 = dateFormat1.format(jDateChooser1.getDate());
String date1 = strDate1.concat(time1);
DateFormat dateFormat2 = new SimpleDateFormat("yyyy-MM-dd ");
String strDate2 = dateFormat2.format(jDateChooser2.getDate());
String date2 = strDate2.concat(time2);
t1 = java.sql.Timestamp.valueOf(date1);
t2 = java.sql.Timestamp.valueOf(date2);
}

public void updateTable(){
try{
Class.forName("com.mysql.cj.jdbc.Driver");
Connection con = DriverManager.getConnection("jdbc:mysql://localhost:3306/travel_agency?zeroDateTimeBehavior=CONVERT_TO_NULL", "root", "root");
String select="SELECT * FROM trip;";
Statement slct = con.createStatement();
ResultSet rs = slct.executeQuery(select);
DefaultTableModel tbModel= (DefaultTableModel) TripTable.getModel();
tbModel.setNumRows(0);
while(rs.next()){
String id = rs.getString("tr_id");
String departure = rs.getString("tr_departure");
String ret = rs.getString("tr_return");
String maxseats = rs.getString("tr_maxseats");
String cost = rs.getString("tr_cost");
String code = rs.getString("tr_br_code");
String gui = rs.getString("tr_gui_AT");
String drv = rs.getString("tr_drv_AT");
String tb_data[] = {id, departure, ret, maxseats, cost, code, gui, drv};
tbModel.addRow(tb_data);
}
con.close();
}catch(ClassNotFoundException | SQLException e){OptionPane.showMessageDialog(this, e.getMessage());}
}

public void updateCombo(){
PickTime1.setText("Set Time");
PickTime2.setText("Set Time");
try{
Class.forName("com.mysql.cj.jdbc.Driver");
Connection con = DriverManager.getConnection("jdbc:mysql://localhost:3306/travel_agency?zeroDateTimeBehavior=CONVERT_TO_NULL", "root", "root");
String select="SELECT br_code FROM branch;";
Statement slct = con.createStatement();
ResultSet rs = slct.executeQuery(select);
DefaultComboBoxModel mod = new DefaultComboBoxModel();
mod.removeAllElements();
mod.addElement("");
while(rs.next()){
int box = rs.getInt("br_code");
mod.addElement(box);
}
tr_br_code.setModel(mod);
con.close();
}catch(ClassNotFoundException | SQLException e){OptionPane.showMessageDialog(this, e.getMessage());}
}

private void PickTime2MouseClicked(java.awt.event.MouseEvent evt) {
timePicker2.showPopup(this, 100, 100);
String time = timePicker2.getSelectedTime();
time=time.replace("MP", "PM");

```

```

        PickTime2.setText(time);
    }

    private void PickTime1MouseClicked(java.awt.event.MouseEvent evt) {
        timePicker1.showPopup(this, 100, 100);
        String time = timePicker1.getSelectedTime();
        time=time.replace("MP", "PM");
        PickTime1.setText(time);
    }

    private void formWindowOpened(java.awt.event.WindowEvent evt) {
        jLabel5.setVisible(false);
        jLabel8.setVisible(false);
        tr_gui_AT.setVisible(false);
        tr_drv_AT.setVisible(false);
        updateCombo();
        updateTable();
    }

    private void TripTableMouseClicked(java.awt.event.MouseEvent evt) {
        DefaultTableModel tbModel= (DefaultTableModel) TripTable.getModel();
        String dep = tbModel.getValueAt(TripTable.getSelectedRow(), 1).toString();
        String ret = tbModel.getValueAt(TripTable.getSelectedRow(), 2).toString();
        String max = tbModel.getValueAt(TripTable.getSelectedRow(), 3).toString();
        String cost = tbModel.getValueAt(TripTable.getSelectedRow(), 4).toString();
        String br_code = tbModel.getValueAt(TripTable.getSelectedRow(), 5).toString();
        String gui = tbModel.getValueAt(TripTable.getSelectedRow(), 6).toString();
        String drv = tbModel.getValueAt(TripTable.getSelectedRow(), 7).toString();
        tr_maxseats.setText(max);
        tr_cost.setText(cost);
        tr_br_code.setSelectedItem(Integer.valueOf(br_code));
        tr_gui_AT.setSelectedItem(gui);
        tr_drv_AT.setSelectedItem(drv);

        String[] parts1 = dep.split(" ");
        String part1s = parts1[0];
        String part2s = parts1[1];

        jDateChooser1.setDate(java.sql.Date.valueOf(part1s));
        PickTime1.setText(part2s);

        String[] parts2 = ret.split(" ");
        String part1e = parts2[0];
        String part2e = parts2[1];
        jDateChooser2.setDate(java.sql.Date.valueOf(part1e));
        PickTime2.setText(part2e);
    }

    private void InsertActionPerformed(java.awt.event.ActionEvent evt) {
        try{
            Class.forName("com.mysql.cj.jdbc.Driver");
            Connection con = DriverManager.getConnection("jdbc:mysql://localhost:3306/travel_agency?zeroDateTimeBehavior=CONVERT_TO_NULL", "root", "root");
            String insert="INSERT INTO trip(tr_id,tr_departure,tr_return,tr_maxseats,tr_cost,tr_br_code,tr_gui_AT,tr_drv_AT) VALUES(null,?,?,?,?);";
            PreparedStatement insrt = con.prepareStatement(insert);
            convertTimeStamp();
            insrt.setTimestamp(1,t1);
            insrt.setTimestamp(2,t2);
            insrt.setInt(3, Integer.parseInt(tr_maxseats.getText()));
            insrt.setFloat(4,Float.parseFloat(tr_cost.getText()));
            insrt.setInt(5, Integer.parseInt(tr_br_code.getSelectedItem().toString()));
            insrt.setString(6, tr_gui_AT.getSelectedItem().toString());
            insrt.setString(7, tr_drv_AT.getSelectedItem().toString());
            insrt.execute();
            updateTable();
            JOptionPane.showMessageDialog(this, "Inserted Succesfully");
        }catch(ClassNotFoundException | SQLException e){JOptionPane.showMessageDialog(this, e.getMessage());}
    }

    private void Delete1ActionPerformed(java.awt.event.ActionEvent evt) {
        try{
            Class.forName("com.mysql.cj.jdbc.Driver");
            Connection con = DriverManager.getConnection("jdbc:mysql://localhost:3306/travel_agency?zeroDateTimeBehavior=CONVERT_TO_NULL", "root",
"root");
            String delete="DELETE FROM trip WHERE tr_id = ?";
            PreparedStatement dlt = con.prepareStatement(delete);
            DefaultTableModel tbModel= (DefaultTableModel) TripTable.getModel();
            dlt.setInt(1, Integer.parseInt(tbModel.getValueAt(TripTable.getSelectedRow(), 0).toString()));

```

```

        dlt.executeUpdate();
        updateTable();
        jDateChooser1.setDate(null);
        jDateChooser2.setDate(null);
        PickTime1.setText("");
        PickTime2.setText("");
        tr_maxseats.setText("");
        tr_cost.setText("");
        JOptionPane.showMessageDialog(this, "Deleted Succesfully");
        con.close();
    }catch(ClassNotFoundException | SQLException e){JOptionPane.showMessageDialog(this, e.getMessage());}
}

private void Update1ActionPerformed(java.awt.event.ActionEvent evt) {
    convertTimeStamp();
    try{
        Class.forName("com.mysql.cj.jdbc.Driver");
        Connection con = DriverManager.getConnection("jdbc:mysql://localhost:3306/travel_agency?zeroDateTimeBehavior=CONVERT_TO_NULL", "root",
"root");
        String update="UPDATE trip SET tr_departure = ?, tr_return = ?, tr_maxseats = ?, tr_cost = ?, tr_br_code = ?, tr_gui_AT = ?, tr_drv_AT = ? WHERE tr_id = ?";
        PreparedStatement upd = con.prepareStatement(update);
        DefaultTableModel tbModel= (DefaultTableModel) TripTable.getModel();
        upd.setTimestamp(1, t1);
        upd.setTimestamp(2, t2);
        upd.setInt(3, Integer.parseInt(tr_maxseats.getText()));
        upd.setFloat(4, Float.parseFloat(tr_cost.getText()));
        upd.setInt(5, Integer.parseInt(tr_br_code.getSelectedItem().toString()));
        upd.setString(6, tr_gui_AT.getSelectedItem().toString());
        upd.setString(7, tr_drv_AT.getSelectedItem().toString());
        upd.setInt(8, Integer.parseInt(tbModel.getValueAt(TripTable.getSelectedRow(), 0).toString()));
        upd.executeUpdate();
        updateTable();
        JOptionPane.showMessageDialog(this, "Updated Succesfully");
        con.close();
    }catch(ClassNotFoundException | SQLException e){JOptionPane.showMessageDialog(this, e.getMessage());}
}

private void CancelActionPerformed(java.awt.event.ActionEvent evt) {
    dispose();
    Main main = new Main();
    main.show();
}

private void tr_br_codeActionPerformed(java.awt.event.ActionEvent evt) {
    try{
        Class.forName("com.mysql.cj.jdbc.Driver");
        Connection con = DriverManager.getConnection("jdbc:mysql://localhost:3306/travel_agency?zeroDateTimeBehavior=CONVERT_TO_NULL", "root", "root");
        DefaultComboBoxModel mod2 = new DefaultComboBoxModel();
        DefaultComboBoxModel mod3 = new DefaultComboBoxModel();
        if(!"".equals(tr_br_code.getSelectedItem().toString())){
            mod2.removeAllElements();
            mod3.removeAllElements();
            jLabel5.setVisible(true);
            jLabel8.setVisible(true);
            tr_gui_AT.setVisible(true);
            tr_drv_AT.setVisible(true);
            String where = tr_br_code.getSelectedItem().toString();
            String select2="SELECT DISTINCT gui_AT FROM guide INNER JOIN worker ON wrk_AT = gui_AT WHERE wrk_br_code = "+where+"";
            try{
                Statement slct2 = con.createStatement();
                ResultSet rs2 = slct2.executeQuery(select2);
                while(rs2.next()){
                    String box2 = rs2.getString("gui_AT");
                    mod2.addElement(box2);
                }
                tr_gui_AT.setModel(mod2);
                String select3="SELECT DISTINCT drv_AT FROM driver INNER JOIN worker ON wrk_AT = drv_AT WHERE wrk_br_code = "+where+"";
                Statement slct3 = con.createStatement();
                ResultSet rs3 = slct3.executeQuery(select3);
                while(rs3.next()){
                    String box3 = rs3.getString("drv_AT");
                    mod3.addElement(box3);
                }
                tr_drv_AT.setModel(mod3);
            }
        }
    }catch(SQLException c){System.out.println(c.getMessage());}
    con.close();
}

```

```
}catch(ClassNotFoundException | SQLException e){OptionPane.showMessageDialog(this, e.getMessage());}
```

```
}
```

## wrk\_mng.java

### Κώδικας:

```
public void updateTable(){
    try{
        Class.forName("com.mysql.cj.jdbc.Driver");
        Connection con = DriverManager.getConnection("jdbc:mysql://localhost:3306/travel_agency?zeroDateTimeBehavior=CONVERT_TO_NULL", "root", "root");
        String select="SELECT * FROM worker;";
        Statement slct = con.createStatement();
        ResultSet rs = slct.executeQuery(select);
        DefaultTableModel tbModel= (DefaultTableModel) WorkerTable.getModel();
        tbModel.setNumRows(0);
        while(rs.next()){
            String at = rs.getString("wrk_AT");
            String name = rs.getString("wrk_name");
            String lname = rs.getString("wrk_lname");
            String salary = rs.getString("wrk_salary");
            String code = Integer.toString(rs.getInt("wrk_br_code"));
            String tb_data[] = {at, name, lname, salary, code};
            tbModel.addRow(tb_data);
        }
        con.close();
    }catch(ClassNotFoundException | SQLException e){OptionPane.showMessageDialog(this, e.getMessage());}
}

public void updateCombo(){
    try{
        Class.forName("com.mysql.cj.jdbc.Driver");
        Connection con = DriverManager.getConnection("jdbc:mysql://localhost:3306/travel_agency?zeroDateTimeBehavior=CONVERT_TO_NULL", "root", "root");
        String select="SELECT br_code FROM branch;";
        Statement slct = con.createStatement();
        ResultSet rs = slct.executeQuery(select);
        DefaultComboBoxModel mod = new DefaultComboBoxModel();
        mod.removeAllElements();
        while(rs.next()){
            String box = "Branch: "+rs.getInt("br_code")+"";
            mod.addElement(box);
        }
        wrk_br_code.setModel(mod);
        con.close();
    }catch(ClassNotFoundException | SQLException e){OptionPane.showMessageDialog(this, e.getMessage());}
}

private void formWindowOpened(java.awt.event.WindowEvent evt) {
    updateTable();
    updateCombo();
}

private void WorkerTableMouseClicked(java.awt.event.MouseEvent evt) {
    DefaultTableModel tbModel= (DefaultTableModel) WorkerTable.getModel();
    String at = tbModel.getValueAt(WorkerTable.getSelectedRow(), 0).toString();
    String name = tbModel.getValueAt(WorkerTable.getSelectedRow(), 1).toString();
    String lname = tbModel.getValueAt(WorkerTable.getSelectedRow(), 2).toString();
    String salary = tbModel.getValueAt(WorkerTable.getSelectedRow(), 3).toString();
    String code = "Branch: "+tbModel.getValueAt(WorkerTable.getSelectedRow(), 4).toString()+"";
    wrk_AT.setText(at);
    wrk_name.setText(name);
    wrk_lname.setText(lname);
    wrk_salary.setText(salary);
    wrk_br_code.setSelectedItem(code);
}

private void Delete1ActionPerformed(java.awt.event.ActionEvent evt) {
    try{
        Class.forName("com.mysql.cj.jdbc.Driver");
        Connection con = DriverManager.getConnection("jdbc:mysql://localhost:3306/travel_agency?zeroDateTimeBehavior=CONVERT_TO_NULL", "root", "root");
        String del= "DELETE FROM worker WHERE wrk_AT = ?;";
        PreparedStatement dlt = con.prepareStatement(del);
        dlt.setString(1, wrk_AT.getText());
        dlt.executeUpdate();
        updateTable();
        wrk_AT.setText("");
        wrk_name.setText("");
        wrk_lname.setText("");
    }
```

```

        wrk_salary.setText("");
        JOptionPane.showMessageDialog(this, "Deleted Succesfully");
        con.close();
    }catch(ClassNotFoundException | SQLException e){JOptionPane.showMessageDialog(this, e.getMessage());}
}

private void Update1ActionPerformed(java.awt.event.ActionEvent evt) {
    try{
        DefaultTableModel tbModel= (DefaultTableModel) WorkerTable.getModel();
        Class.forName("com.mysql.cj.jdbc.Driver");
        Connection con = DriverManager.getConnection("jdbc:mysql://localhost:3306/travel_agency?zeroDateTimeBehavior=CONVERT_TO_NULL", "root", "root");
        String update="UPDATE worker SET wrk_name = ?, wrk_lname = ?, wrk_salary = ?, wrk_br_code = ? WHERE wrk_AT = ?";
        PreparedStatement upd = con.prepareStatement(update);
        String code = wrk_br_code.getSelectedItem().toString().replaceAll("[^0-9]", "");
        int i=Integer.parseInt(code);
        upd.setString(1, wrk_name.getText());
        upd.setString(2, wrk_lname.getText());
        upd.setFloat(3, Float.parseFloat(wrk_salary.getText()));
        upd.setInt(4, i);
        upd.setString(5, tbModel.getValueAt(WorkerTable.getSelectedRow(), 0).toString());
        upd.executeUpdate();
        updateTable();
        JOptionPane.showMessageDialog(this, "Updated Succesfully");
        con.close();
    }catch(ClassNotFoundException | SQLException e){JOptionPane.showMessageDialog(this, e.getMessage());}
}

private void CancelActionPerformed(java.awt.event.ActionEvent evt) {
    dispose();
    Main main = new Main();
    main.show();
}

private void InsertActionPerformed(java.awt.event.ActionEvent evt) {
    try{
        Class.forName("com.mysql.cj.jdbc.Driver");
        Connection con = DriverManager.getConnection("jdbc:mysql://localhost:3306/travel_agency?zeroDateTimeBehavior=CONVERT_TO_NULL", "root", "root");
        String insert="INSERT INTO worker(wrk_AT,wrk_name,wrk_lname,wrk_salary,wrk_br_code) VALUES(?,?,?,?,?)";
        PreparedStatement insrt = con.prepareStatement(insert);
        String code = wrk_br_code.getSelectedItem().toString().replaceAll("[^0-9]", "");
        int i=Integer.parseInt(code);
        insrt.setString(1,wrk_AT.getText());
        insrt.setString(2,wrk_name.getText());
        insrt.setString(3,wrk_lname.getText());
        insrt.setFloat(4, Float.parseFloat(wrk_salary.getText()));
        insrt.setInt(5,i);
        insrt.executeUpdate();
        updateTable();
        JOptionPane.showMessageDialog(this, "Inserted Succesfully");
    }catch(ClassNotFoundException | SQLException e){JOptionPane.showMessageDialog(this, e.getMessage());}
}
}

```

## Επεξήγηση Κώδικα:

Οι έξτρα λειτουργίες της εφαρμογής μας, αφορούν και το εμφανισιακό κομμάτι, αλλά και το λειτουργικό κομμάτι. Συγκεκριμένα, όπως αναφέρθηκε και στην εισαγωγή, χρησιμοποιήσαμε κάποια εκπαιδευτικά βίντεο στην Java Swing ώστε να δημιουργήσουμε ένα όμορφο και ελκυστικό στον χρήστη Menu (μέσω των αρχείων ModelMenu.java, MenuManager.java, MenuItem.java, custom jTable (μέσω του αρχείου Table.java, Table1.java, TableHeader.java, TableStatus.java), custom JComboBox (μέσω του αρχείου CustomJCombo.java), custom JButtons (μέσω του αρχείου CustomButton.java), custom ScrollBar (μέσω του αρχείου ModernScrollBar.java και ScrollBar.java).

Για τις Extra λειτουργίες, φροντίσαμε να δημιουργήσουμε μια νέα διεπαφή για τον Manager του κάθε υποκαταστήματος. Αυτό προϋπόθετε κάποιες αλλαγές στον πίνακα manages καθώς χρειαζόμασταν Usernames και Passwords για αυτούς τους χρήστες, όπως αναφέρθηκε και στο 1<sup>ο</sup> κεφάλαιο.

Συνεχίζοντας, θέλαμε ο Manager να μην έχει τα ίδια δικαιώματα με τον IT Manager και γι' αυτό τον λόγο δημιουργήσαμε καινούρια αρχεία .java για τον manages. Συγκεκριμένα, οι λειτουργίες που μπορεί να εκτελέσει είναι να προβάλλει τους υπαλλήλους και τους μισθούς (καθώς και τη συνολική μισθοδοσία) στο Branch που διευθύνει, καθώς και να προβάλλει, αλλάξει και διαγράψει τα δεδομένα των πινάκων Worker, Driver, Admin, Guide και Trip για το Branch το οποίο διευθύνει. Αυτά τα αρχεία έχουν παρόμοια δομή με του IT Manager, οπότε δεν κρίνεται σκόπιμο να τα παρουσιάσουμε ξανά. Παραθέτουμε παρακάτω Screenshot με το Menu του Manager:

Trip ID	Trip Departure	Trip Return	Trip Cost	Branch Code	Status
1	2023-01-29 11:00:00	2023-02-08 18:00:00	1023.11	1	Available_Seats
11	2023-05-23 11:00:00	2023-05-30 17:30:00	858.31	1	Available_Seats

Έπειτα, δημιουργήσαμε τα λεγόμενα cards (μέσω των αρχείων Model\_Card.java και Card.java) που εμφανίζονται στον IT Manager και στον Manager και προσφέρουν χρήσιμες πληροφορίες στο Homepage των χρηστών. Συγκεκριμένα, για τον IT Manager εμφανίζει το συνολικό ποσό εσόδων και εξόδων για όλα τα Branches και την επόμενη εκδρομή που είναι προγραμματισμένη. Για τον Manager, εμφανίζεται το συνολικό ποσό εσόδων και εξόδων, τον αριθμό των εργαζομένων και την επόμενη εκδρομή για το Branch που διευθύνει. Παρακάτω είναι τα παραδείγματα των Cards για τους δύο χρήστες.

### IT Manager:

Total Profit	Total Costs	Upcoming Trip	Number Of Employees
41821.62€	53981.10€	2023-01-27 12:30:00	8
Total Profit From all Branches	Total Costs From all Branches	Next trip on Calendar	Number of Employees in Branch 1

### Manager:

Total Profit	Total Costs	Upcoming Trip	Number Of Employees
4785.95€	15394.30€	2023-01-29 11:00:00	8
Total Profit From Branch 1	Total Costs From Branch 1	Next trip on Calendar	Number of Employees in Branch 1



Τέλος, στην αρχική σελίδα των IT Manager και Manager δημιουργήσαμε ένα πινακάκι το οποίο δείχνει τα ταξίδια όλων των branches και του branch το οποίο διευθύνει αντίστοιχα. Στο τελευταίο κελί, θεωρήσαμε χρήσιμο να υπάρχει επισήμανση για το Status των κρατήσεων σε κάθε διαδρομή, καθώς εμφανίζονται τα παρακάτω:

- ❖ Αν δεν υπάρχουν διαθέσιμες θέσεις: Fully\_Booked
- ❖ Αν υπάρχουν μερικές διαθέσιμες θέσεις (πάνω από το 50% πληρότητα): Limited\_Availability
- ❖ Αν υπάρχουν πολλές διαθέσιμες θέσεις (κάτω από το 50% πληρότητα): Fully\_Booked

Pending Trips

Trip ID	Trip Departure	Trip Return	Trip Cost	Brach Code	Status
14	2023-04-01 11:30:00	2023-04-08 21:00:00	570.99	4	Available_Seats
5	2023-05-09 11:30:00	2023-05-20 17:30:00	822.97	10	Available_Seats
11	2023-05-23 11:00:00	2023-05-30 17:30:00	858.31	1	Available_Seats
4	2023-06-03 12:30:00	2023-06-15 16:00:00	842.54	4	Available_Seats
15	2023-07-22 12:30:00	2023-07-28 19:30:00	700.99	10	Limited_Availability
6	2023-08-13 11:30:00	2023-08-23 18:00:00	677.91	6	Available_Seats
16	2023-09-09 11:30:00	2023-09-15 18:30:00	790.89	6	Fully_Booked
7	2023-09-12 12:30:00	2023-09-17 19:30:00	532.12	3	Available_Seats
17	2023-09-15 10:30:00	2023-09-19 19:30:00	350.99	3	Available_Seats

## Βιβλιογραφία

- ❖ Tutorial Video Part 1: <http://bit.ly/3XnNR9d>
- ❖ Tutorial Video Part 2: <http://bit.ly/3kaw330>
- ❖ Tutorial Video Part 3: <http://bit.ly/3iFc77X>
- ❖ GitHub Link for Custom Scroll Bar, JButton, JComboBox: <https://github.com/DJ-Raven/raven-project>
- ❖ Website Link for Custom Date Chooser: <https://toedter.com/jcalendar>
- ❖ GitHub Link for Custom Time Chooser: <https://github.com/DJ-Raven/java-swing-timepicker>