

Project Εργαστηρίου Βάσεων Δεδομένων

Περιεχόμενα

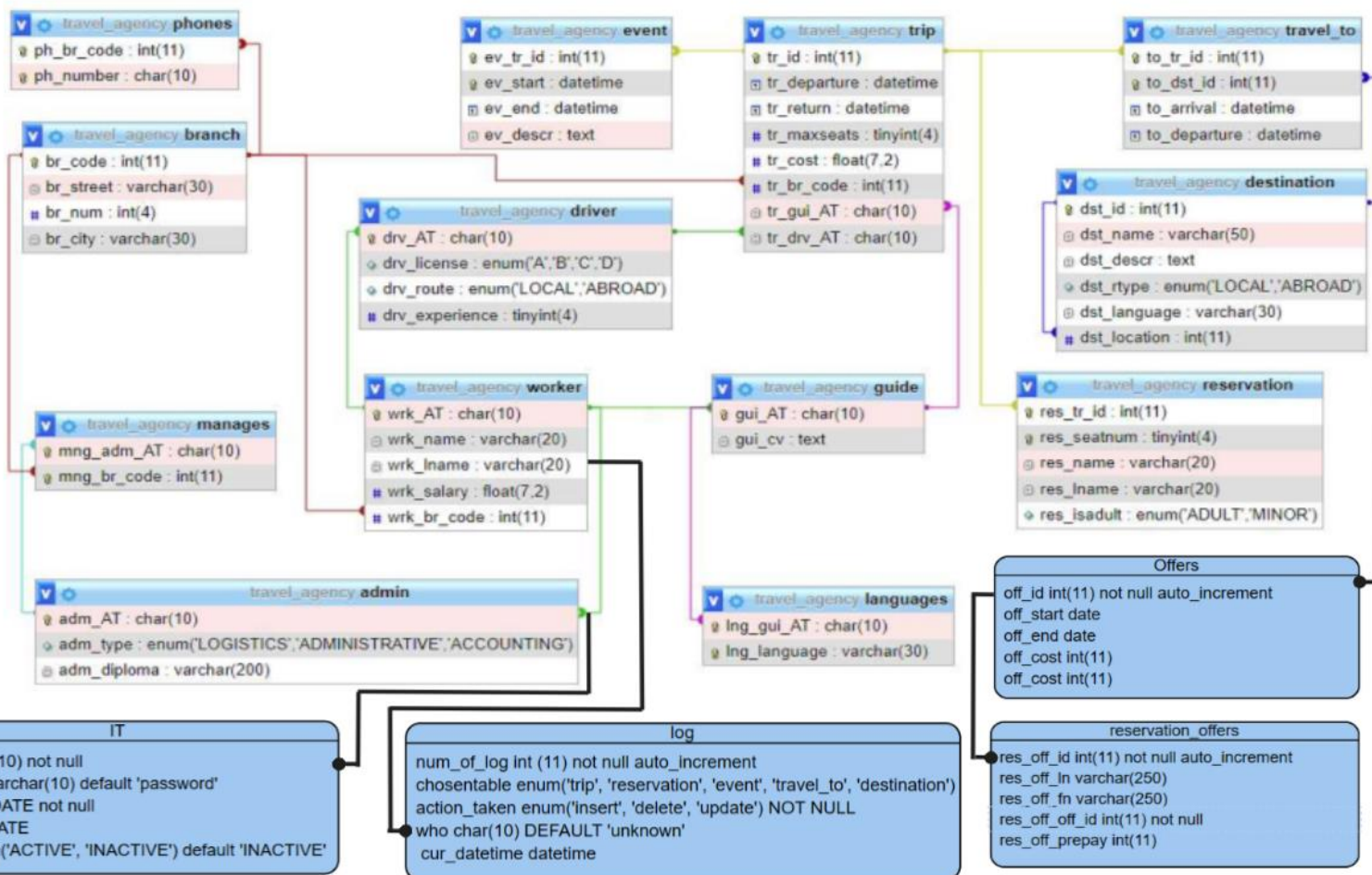
1.Παραδοχές.....	3
2.Περιγραφή της Βάσης Δεδομένων & Διάγραμμα Βάσης.....	4
3. Περιεχόμενο Εργασίας	6
3.1 Μέρος Α: Σχεδιασμός ΒΔ και SQL.....	6
3.1.2 Δημιουργία Stored Procedure.....	6
3.1.3 Δημιουργία Trigger.....	14
3.2 Μέρος Β: GUI.....	23
4. Σενάριο.....	27
5. Extra λειτουργία.....	35
6. Κώδικες.....	37

1. ΠΑΡΑΔΟΧΕΣ

1. trigger_reservation: δημιουργήσαμε επιπλέον trigger από αυτούς που ζητούνται ώστε όταν γίνεται κράτηση στον πίνακα reservation τότε το πεδίο seatnum να παίρνει τιμή η οποία κάθε φορά θα είναι η επόμενη θέση από την προηγούμενη κράτηση. Κατά αυτόν τον τρόπο οι θέσεις θα έχουν αύξουσα διαδοχική σειρά.
2. Πίνακας log: ορίσαμε επιπλέον πίνακα στη βάση για να καταγράφεται η δραστηριότητα των υπεύθυνων πληροφορικής της εταιρίας για τους πίνακες trip, reservation, event, travel_to, destination. Εξηγείται λεπτομερώς παρακάτω.
3. Οι εντολές insert για την δημιουργία της βάσης έχουν παραχθεί μέσω προγράμματος python. Το αρχείο βρίσκεται μέσα στο zip με την ονομασία database_final.py . Αν επιθυμείτε να το εκτελέσετε για να παράγετε τις insert εντολές θα πρέπει να αλλάξετε την τιμή της μεταβλητής path1. Θα πρέπει να επιλέξετε σε ποιο path θέλετε να παραχθεί το αρχείο και στο ίδιο path που θα επιλέξετε για να παραχθεί θα πρέπει να υπάρχει το αρχείο names.txt (αυτό είναι το txt που μας δίνετε το οποίο περιέχει ονόματα).
4. Παραλείψαμε τις εντολές insert στον πίνακα reservation_offers λόγω χώρου.

2.Περιγραφή της Βάσης Δεδομένων & Διάγραμμα Βάσης

Σχεσιακό διάγραμμα της συνολικής αναθεωρημένης ΒΔ



Προσθέσαμε 4 νέους πίνακες:

1. Offers: προσφορές ταξιδιών
2. Reservation_offers: οι κρατήσεις για τις προσφορές των ταξιδιών
3. IT: υπεύθυνος πληροφορικής του πρακτορείου, έχουμε προσθέσει πεδίο με ονομασία active που δηλώνει το status το υπεύθυνου πληροφορικής ώστε να γνωρίζουμε κάθε φορά ποιος IT κάνει τις αλλαγές στη βάση και να καταχωρούμε το όνομά του στον πίνακα log.
4. Log: πίνακας που αποθηκεύεται όλη η δραστηριότητα. Κάθε ενέργεια εισαγωγής, ενημέρωσης ή διαγραφής στους πίνακες trip, reservation, event, travel_to, destination, καθώς και το ποιος την εκτέλεσε.

Για τις ανάγκες του project επιλέξαμε να προσθέσουμε επιπλέον τον πίνακα log με τον οποίο αποθηκεύουμε τις μετατροπές που κάνουν οι υπεύθυνοι πληροφορικής του πρακτορείου. Ο πίνακας log αποτελείται από τα εξής πεδία:

- num_of_log int (11) not null auto_increment: αριθμός ο οποίος δίνεται σε κάθε ενέργεια και είναι μοναδικός, αυξάνεται αυτόματα και είναι το κλειδί του πίνακα.
- chosentable enum('trip', 'reservation', 'event', 'travel_to', 'destination'): το πεδίο δηλώνει τον πίνακα στον οποίο έκανε αλλαγή ο υπεύθυνος πληροφορικής(IT).
- action_taken enum('insert', 'delete', 'update') NOT NULL: δηλώνει το είδος της ενέργειας.
- who char(10) DEFAULT 'unknown': ποιος έκανε την ενέργεια, το επώνυμο του υπεύθυνου πληροφορικής που την έκανε.
- cur_datetime datetime: η τρέχουσα ώρα και ημέρα πραγματοποίηση της ενέργειας.

Τα πεδία των υπόλοιπων πινάκων έχουν σχεδιαστεί σύμφωνα με την εκφώνηση και δεν έχουμε κάνει καμία προσθήκη δική μας.

Οι εντολές create και insert για τη δημιουργία της βάσης βρίσκονται στα αρχεία του zip με ονομασία “1.create table.txt” και “3.automation.txt” αντίστοιχα και στο τέλος της αναφοράς τα παραθέτουμε όλα για οικονομία χώρου.

3. Περιεχόμενο Εργασίας

3.1.2 Δημιουργία Stored Procedure

- Ερώτημα 3.1.3.1

drop procedure if exists InsertDriver;

delimiter \$

create procedure InsertDriver(in AT char(10), in name varchar(20), in lastname varchar(20), in salary float(7,2),
in license enum('A','B','C','D'), in route enum('LOCAL','ABROAD'), in experience tinyint(4))

BEGIN

declare branch_code int(11);

SELECT wrk_br_code into branch_code from worker inner join driver on wrk_AT = drv_AT GROUP BY
wrk_br_code ORDER BY count(*) limit 1;

insert into worker values(AT, name, lastname, salary, branch_code);

insert into driver values(AT, license, route, experience);

END\$

DELIMITER ;

call InsertDriver('AM2754238','Giorgos','Samios', 5000, 'B', 'LOCAL', 35);

Στην παρακάτω εικόνα είναι ένα παράδειγμα για την κλήση της procedure InsertDriver:

Κάνουμε select στην ένωση των πινάκων driver και worker για να εμφανίσει πόσους οδηγούς έχει κάθε επιχείρηση και να βρούμε αυτή με τους λιγότερους. Έπειτα καλούμε την procedure που δημιουργήσαμε και ο οδηγός που έχουμε εισάγει βρίσκεται στην επιχείρηση με τους λιγότερους.

```
mysql> SELECT wrk_br_code, count(*) from worker inner join driver on wrk_AT = drv_AT GROUP BY wrk_br_code;
+-----+-----+
| wrk_br_code | count(*) |
+-----+-----+
|          6 |         2 |
|          8 |         2 |
|          7 |         2 |
|          3 |         2 |
|          9 |         2 |
|          5 |         2 |
|          4 |         2 |
|          2 |         2 |
|         10 |         1 |
+-----+-----+
9 rows in set (0.00 sec)

mysql> call InsertDriver('AM2754237','Giorgos','Samios', 5000, 'B', 'LOCAL', 35);
Query OK, 1 row affected (0.01 sec)

mysql> SELECT wrk_br_code, count(*) from worker inner join driver on wrk_AT = drv_AT GROUP BY wrk_br_code;
+-----+-----+
| wrk_br_code | count(*) |
+-----+-----+
|          6 |         2 |
|          8 |         2 |
|          7 |         2 |
|          3 |         2 |
|          9 |         2 |
|          5 |         2 |
|          4 |         2 |
|         10 |         2 |
|          2 |         2 |
+-----+-----+
9 rows in set (0.00 sec)
```

- Ερώτημα 3.1.3.2

drop procedure if exists SearchTrip;

delimiter \$

create procedure SearchTrip(in id int, in startdate datetime, in enddate datetime)

BEGIN

```
CREATE TEMPORARY TABLE temp1 (tr_id1 INT,tr_cost INT,tr_maxseats INT,  
    wrk_name varchar(20), wrk_lname varchar(20), tr_departure DATETIME,  
    tr_return DATETIME);
```

```
CREATE TEMPORARY TABLE temp2 (tr_id2 INT,numofreservations INT,availableseats INT);
```

```
insert into temp1 select tr_id as trip1, tr_cost,tr_maxseats,wrk_name,wrk_lname,  
tr_departure, tr_return
```

```
from branch inner join trip on tr_br_code=br_code
```

```
    inner join driver on drv_AT=tr_drv_AT
```

```
    inner join guide on tr_gui_AT=gui_AT
```

```
    inner join worker on wrk_AT=gui_AT or wrk_AT=drv_AT
```

```
where br_code=id and startdate>=tr_departure and startdate<tr_return
```

```
and enddate>tr_departure and enddate<=tr_return;
```

```
insert into temp2 select tr_id as trip2, count(*) as numofreservations , tr_maxseats-  
count(*) as availableseats
```

```
from branch
```

```
    inner join trip on br_code=tr_br_code
```

```
    inner join reservation on tr_id=res_tr_id
```

```
where br_code=id
```

```
group by res_tr_id;
```

```
select * from temp1 inner join temp2 on tr_id1=tr_id2;
```

```
drop table temp1;
```



```
drop table temp2;
```

```
END$
```

```
DELIMITER ;
```

```
call SearchTrip(4, '2022-11-13 09:00:00', '2022-12-16 12:00:00');
```

Δίνουμε ως όρισμα τον αριθμό του υποκαταστήματος και 2 ημερομηνίες και επιστρέφονται πληροφορίες για το ταξίδι:

Κωδικός ταξιδιού, κόστος, μέγιστες θέσεις, όνομα ξεναγού και οδηγού, επώνυμο ξεναγού και οδηγού, ημερομηνία αναχώρησης και επιστροφής, κωδικός ταξιδιού ξανά, αριθμός κρατήσεων για κάθε ταξίδι και διαθέσιμες θέσεις.

(τα δύο τελευταία στοιχεία αναφέρονται δύο φορές διότι για τις ανάγκες του gui έπρεπε να τα συμπτύξουμε όλα σε ένα πίνακα)

```
call SearchTrip(4, '2022-11-13 09:00:00', '2022-12-16 12:00:00');
```

tr_id1	tr_cost	tr_maxseats	wrk_name	wrk_lname	tr_departure	tr_return	tr_id2	numofreservations	availableseats
4	900	33	Aaren	White	2022-11-13 09:00:00	2022-12-16 12:00:00	4	10	23
4	900	33	Lachish	Davis	2022-11-13 09:00:00	2022-12-16 12:00:00	4	10	23
14	900	27	Aaren	White	2022-11-13 09:00:00	2022-12-16 12:00:00	14	10	17
14	900	27	Lachish	Davis	2022-11-13 09:00:00	2022-12-16 12:00:00	14	10	17

```
4 rows in set (0.00 sec)
```

```
Query OK, 0 rows affected (0.02 sec)
```

- Ερώτημα 3.1.3.3

drop procedure if exists deleteAdmin;

delimiter \$

create procedure deleteAdmin(in name varchar(20), in lname varchar(20))

BEGIN

DECLARE type enum('LOGISTICS','ADMINISTRATIVE','ACCOUNTING');

select adm_type into type

from worker

inner join admin on wrk_AT=adm_AT

where wrk_name=name and wrk_lname=lname;

if (type = 'ADMINISTRATIVE') then

SIGNAL SQLSTATE VALUE '45000'

SET MESSAGE_TEXT = 'CAN NOT DELETE ADMIN.';

end if;

delete from worker where wrk_name=name and wrk_lname=lname;

END\$

DELIMITER ;

call deleteAdmin('Nadiya', 'Thompson');

Δίνουμε ως όρισμα το όνομα και το επώνυμο ενός υπαλλήλου ο οποίος είναι διευθυντής και δεν τον διαγράφει ενώ όταν δώσουμε το όνομα και το επώνυμο ενός υπαλλήλου που είναι διοικητικός τότε τον διαγράφει.

```
mysql> call deleteAdmin( 'Nadiya', 'Thompson');  
ERROR 1644 (45000): CAN NOT DELETE ADMIN.  
mysql> |
```

```
mysql> call deleteAdmin( 'Creigh', 'Moore');  
Query OK, 1 row affected (0.01 sec)
```

- Ερώτημα 3.1.3.4.α

drop procedure if exists customerPrepay;

delimiter \$

create procedure customerPrepay (in minimumvalue int, in maximumvalue int)

BEGIN

select res_off_ln, res_off_fn from reservation_offers

where res_off_prepay BETWEEN minimumvalue and maximumvalue;

END\$

DELIMITER ;

call customerPrepay(50,200);

Δημιουργία INDEXING για το procedure customerPrepay:

CREATE INDEX Prepay USING HASH ON reservation_offers (res_off_prepay);

DROP INDEX Prepay ON reservation_offers;

Αποτέλεσμα χωρίς index call customerPrepay(50,53);

Russell	Alisa
Wong	Arielle
Bates	Allyson
Willis	Rylan
Lloyd	Alessandra
Sandoval	Allie
Abbott	Moshe
Sharp	Humberto
Brock	Ashleigh

1600 rows in set (0.03 sec)

Query OK, 0 rows affected (0.97 sec)

Αποτέλεσμα με index call customerPrepay(50,53);

Shaw	Sullivan
Cooley	Lucia
Dorsey	Belinda
Mathis	Carter
Olsen	Kelsey
Terrell	Sophie
Lloyd	Alessandra
Sharp	Humberto

1600 rows in set (0.01 sec)

Query OK, 0 rows affected (0.97 sec)

- Ερώτημα 3.1.3.4.β

drop procedure if exists customerReservation;

delimiter \$

create procedure customerReservation (in lastname varchar(250))

BEGIN

select res_off_ln, res_off_fn, res_off_off_id, count(*) from reservation_offers

where res_off_ln=lastname group by res_off_off_id;

END\$

DELIMITER ;

call customerReservation('Ruiz');

Δημιουργία INDEXING για το procedure customerReservation:

CREATE INDEX Reservations USING HASH ON reservation_offers (res_off_ln);

DROP INDEX Reservations ON reservation_offers;

```
mysql> call customerReservation('Higgins');
+-----+-----+-----+-----+
| res_off_ln | res_off_fn | res_off_off_id | count(*) |
+-----+-----+-----+-----+
| Higgins   | Dominik   | 1              | 25       |
| Higgins   | Jayson    | 2              | 34       |
| Higgins   | Armani    | 3              | 31       |
+-----+-----+-----+-----+
3 rows in set (0.12 sec)

Query OK, 0 rows affected (0.12 sec)

mysql> CREATE INDEX Reservations USING HASH ON reservation_offers (res_off_ln);
Query OK, 0 rows affected, 1 warning (0.32 sec)
Records: 0 Duplicates: 0 Warnings: 1

mysql> call customerReservation('Higgins');
+-----+-----+-----+-----+
| res_off_ln | res_off_fn | res_off_off_id | count(*) |
+-----+-----+-----+-----+
| Higgins   | Dominik   | 1              | 25       |
| Higgins   | Jayson    | 2              | 34       |
| Higgins   | Armani    | 3              | 31       |
+-----+-----+-----+-----+
3 rows in set (0.00 sec)

Query OK, 0 rows affected (0.01 sec)
```

3.1.3 Δημιουργία Trigger

- **Ερώτημα 3.1.4.1**

```
DROP TRIGGER IF EXISTS triggerTripInsert;
```

```
DELIMITER $
```

```
CREATE TRIGGER triggerTripInsert AFTER INSERT ON trip
```

```
FOR EACH ROW
```

```
BEGIN
```

```
    declare lastname varchar(20);
```

```
    select wrk_lname into lastname from worker inner join IT on wrk_AT = IT_AT
```

```
    where active='ACTIVE';
```

```
    insert into log values(NULL,'trip', 'insert', lastname, CURRENT_TIMESTAMP());
```

```
END$
```

```
DELIMITER ;
```

```
DROP TRIGGER IF EXISTS triggerTripDelete;
```

```
DELIMITER $
```

```
CREATE TRIGGER triggerTripDelete AFTER DELETE ON trip
```

```
FOR EACH ROW
```

```
BEGIN
```

```
    declare lastname varchar(20);
```

```
    select wrk_lname into lastname from worker inner join IT on wrk_AT = IT_AT
```

```
    where active='ACTIVE';
```

```
    insert into log values(NULL,'trip', 'delete', lastname, CURRENT_TIMESTAMP());
```

```
END$
```

```
DELIMITER ;
```

```
DROP TRIGGER IF EXISTS triggerTripUpdate;
```

```
DELIMITER $
```

```
CREATE TRIGGER triggerTripUpdate AFTER UPDATE ON trip
```

```
FOR EACH ROW
```

```
BEGIN
```

```
    declare lastname varchar(20);
```

```
    select wrk_lname into lastname from worker inner join IT on wrk_AT = IT_AT
```

```
    where active='ACTIVE';
```

```
    insert into log values(NULL,'trip', 'update', lastname, CURRENT_TIMESTAMP());
```

```
END$
```

```
DELIMITER ;
```

```
DROP TRIGGER IF EXISTS triggerReservationInsert;
```

```
DELIMITER $
```

```
CREATE TRIGGER triggerReservationInsert AFTER INSERT ON reservation
```

```
FOR EACH ROW
```

```
BEGIN
```

```
    declare lastname varchar(20);
```

```
    select wrk_lname into lastname from worker inner join IT on wrk_AT = IT_AT
```

```
    where active='ACTIVE';
```

```
    insert into log values(NULL,'reservation', 'insert', lastname, CURRENT_TIMESTAMP());
```

```
END$
```

```
DELIMITER ;
```

```
DROP TRIGGER IF EXISTS triggerReservationDelete;
```

```
DELIMITER $
```

```
CREATE TRIGGER triggerReservationDelete AFTER DELETE ON reservation
```

```
FOR EACH ROW
```

```
BEGIN
```

```
    declare lastname varchar(20);
```

```
    select wrk_lname into lastname from worker inner join IT on wrk_AT = IT_AT
```

```
    where active='ACTIVE';
```

```
    insert into log values(NULL,'reservation', 'delete', lastname, CURRENT_TIMESTAMP());
```

```
END$
```

```
DELIMITER ;
```

```
DROP TRIGGER IF EXISTS triggerReservationUpdate;
```

```
DELIMITER $
```

```
CREATE TRIGGER triggerReservationUpdate AFTER UPDATE ON reservation
```

```
FOR EACH ROW
```

```
BEGIN
```

```
    declare lastname varchar(20);
```

```
    select wrk_lname into lastname from worker inner join IT on wrk_AT = IT_AT
```

```
    where active='ACTIVE';
```

```
    insert into log values(NULL,'reservation', 'update', lastname, CURRENT_TIMESTAMP());
```

```
END$
```

```
DELIMITER ;
```



```
DROP TRIGGER IF EXISTS triggerTravel_toInsert;
```

```
DELIMITER $
```

```
CREATE TRIGGER triggerTravel_toInsert AFTER INSERT ON travel_to
```

```
FOR EACH ROW
```

```
BEGIN
```

```
    declare lastname varchar(20);
```

```
    select wrk_lname into lastname from worker inner join IT on wrk_AT = IT_AT
```

```
    where active='ACTIVE';
```

```
    insert into log values(NULL,'travel_to', 'insert', lastname, CURRENT_TIMESTAMP());
```

```
END$
```

```
DELIMITER ;
```

```
DROP TRIGGER IF EXISTS triggerTravel_toDelete;
```

```
DELIMITER $
```

```
CREATE TRIGGER triggerTravel_toDelete AFTER DELETE ON travel_to
```

```
FOR EACH ROW
```

```
BEGIN
```

```
    declare lastname varchar(20);
```

```
    select wrk_lname into lastname from worker inner join IT on wrk_AT = IT_AT
```

```
    where active='ACTIVE';
```

```
    insert into log values(NULL,'travel_to', 'delete', lastname, CURRENT_TIMESTAMP());
```

```
END$
```

```
DELIMITER ;
```

```
DROP TRIGGER IF EXISTS triggerTravel_toUpdate;
```

```
DELIMITER $
```

```
CREATE TRIGGER triggerTravel_toUpdate AFTER UPDATE ON travel_to
```

```
FOR EACH ROW
```

```
BEGIN
```

```
    declare lastname varchar(20);
```

```
    select wrk_lname into lastname from worker inner join IT on wrk_AT = IT_AT
```

```
    where active='ACTIVE';
```

```
    insert into log values(NULL,'travel_to', 'update', lastname, CURRENT_TIMESTAMP());
```

```
END$
```

```
DELIMITER ;
```

```
DROP TRIGGER IF EXISTS triggerDestinationInsert;
```

```
DELIMITER $
```

```
CREATE TRIGGER triggerDestinationInsert AFTER INSERT ON destination
```

```
FOR EACH ROW
```

```
BEGIN
```

```
    declare lastname varchar(20);
```

```
    select wrk_lname into lastname from worker inner join IT on wrk_AT = IT_AT
```

```
    where active='ACTIVE';
```

```
    insert into log values(NULL,'destination', 'insert', lastname, CURRENT_TIMESTAMP());
```

```
END$
```

```
DELIMITER ;
```

```
DROP TRIGGER IF EXISTS triggerDestinationDelete;
```

```
DELIMITER $
```

```
CREATE TRIGGER triggerDestinationDelete AFTER DELETE ON destination
```

```
FOR EACH ROW
```

```
BEGIN
```

```
    declare lastname varchar(20);
```

```
    select wrk_lname into lastname from worker inner join IT on wrk_AT = IT_AT
```

```
    where active='ACTIVE';
```

```
    insert into log values(NULL,'destination', 'delete', lastname, CURRENT_TIMESTAMP());
```

```
END$
```

```
DELIMITER ;
```

```
DROP TRIGGER IF EXISTS triggerDestinationUpdate;
```

```
DELIMITER $
```

```
CREATE TRIGGER triggerDestinationUpdate AFTER UPDATE ON destination
```

```
FOR EACH ROW
```

```
BEGIN
```

```
    declare lastname varchar(20);
```

```
    select wrk_lname into lastname from worker inner join IT on wrk_AT = IT_AT
```

```
    where active='ACTIVE';
```

```
    insert into log values(NULL,'destination', 'update', lastname, CURRENT_TIMESTAMP());
```

```
END$
```

```
DELIMITER ;
```

Πραγματοποιούμε select για τον πίνακα log ώστε να δούμε ότι έχουν καταγραφεί οι ενέργειες που έχουμε κάνει:

```
mysql> select * from log;
```

num_of_log	chosentable	action_taken	who	cur_datetime
1	trip	insert	Smith	2023-01-20 17:56:23
297	trip	update	Brown	2023-01-27 19:04:46
298	trip	delete	Brown	2023-01-27 19:04:50
299	trip	insert	Brown	2023-01-27 19:04:52
300	travel_to	update	Brown	2023-01-27 19:05:16
301	travel_to	delete	Brown	2023-01-27 19:05:19
302	travel_to	insert	Brown	2023-01-27 19:05:22
303	reservation	update	Brown	2023-01-27 19:05:49
304	reservation	delete	Brown	2023-01-27 19:05:55
305	reservation	insert	Brown	2023-01-27 19:05:58
306	destination	update	Brown	2023-01-27 19:06:15
307	destination	delete	Brown	2023-01-27 19:06:18
308	destination	insert	Brown	2023-01-27 19:06:20

- Ερώτημα 3.1.4.2

```
DROP TRIGGER IF EXISTS trigger2;
```

```
DELIMITER $
```

```
CREATE TRIGGER trigger2 BEFORE UPDATE ON trip
```

```
FOR EACH ROW
```

```
BEGIN
```

```
    declare counter1 int;
```

```
    select count(*) into counter1 from reservation inner join trip on tr_id=res_tr_id
```

```
    where tr_id=new.tr_id group by tr_id;
```

```
    if (counter1>0 and (old.tr_departure<>new.tr_departure or old.tr_return<>new.tr_return  
    or old.tr_cost<>new.tr_cost)) then
```

```
        SIGNAL SQLSTATE VALUE '45000'
```

```
        SET MESSAGE_TEXT = 'CAN NOT UPDATE';
```

```
    end if;
```

```
END$
```

```
DELIMITER ;
```

Προσπαθούμε να αλλάξουμε το κόστος του ταξιδιού για το ταξίδι με κωδικό 4 και επειδή όπως βλέπουμε παρακάτω υπάρχουν κρατήσεις δεν μας επιτρέπει.

```
mysql> UPDATE trip
-> SET tr_cost=250
-> WHERE tr_id=4;
ERROR 1644 (45000): CAN NOT UPDATE
mysql> select * from reservation where res_tr_id=4;
```

res_tr_id	res_seatnum	res_name	res_lname	res_isadult
4	1	Lemcke	Mcdonald	ADULT
4	2	Zamir	Smith	ADULT
4	5	Fillander	Butler	MINOR
4	6	Henden	Olson	ADULT
4	7	Leander	Richardson	ADULT
4	8	Hyoze	Bailey	ADULT
4	9	Foskett	Bennett	MINOR
4	10	Shanleigh	Robertson	MINOR
4	11	Eisenstark	Lee	ADULT
4	12	Ulda	Chavez	ADULT

```
10 rows in set (0.00 sec)
```

- Ερώτημα 3.1.4.3

```
DROP TRIGGER IF EXISTS trigger3;
```

```
DELIMITER $
```

```
CREATE TRIGGER trigger3 BEFORE UPDATE ON worker
```

```
FOR EACH ROW
```

```
BEGIN
```

```
    If (new.wrk_salary < old.wrk_salary) then
```

```
        SIGNAL SQLSTATE VALUE '45000'
```

```
        SET MESSAGE_TEXT = 'CAN NOT REDUCE WORKER SALARY!';
```

```
    end if;
```

```
END$
```

```
DELIMITER ;
```

Εμφανίζουμε αρχικά τον μισθό του υπαλλήλου με το παρακάτω AT και έπειτα κάνουμε ενημέρωση για τον συγκεκριμένο υπάλληλο με μισθό μικρότερο. Ο trigger δεν μας το επιτρέπει.

```
mysql> select * from worker where wrk_AT = 'AM2571561';
+-----+-----+-----+-----+-----+
| wrk_AT | wrk_name | wrk_lname | wrk_salary | wrk_br_code |
+-----+-----+-----+-----+-----+
| AM2571561 | Aarika | Thompson | 3000.00 | 2 |
+-----+-----+-----+-----+-----+
1 row in set (0.00 sec)

mysql> update worker set wrk_salary = 2000.00 where wrk_AT = 'AM2571561';
ERROR 1644 (45000): CAN NOT REDUCE WORKER SALARY!
mysql> |
```

3.2 Μέρος B: GUI

- Ερώτημα 3.2.1 και 3.2.2

Περιγραφή gui:

public GetConnection(){...}

Η κλάση αυτή χρησιμοποιείται για να συνδεθούμε με τη βάση μας. Χρησιμοποιείται σε όλες τις κλάσεις που δημιουργούμε για να εκτελούμε τα queries. Θα πρέπει να έχουν συμπληρωθεί τα πεδία DB_URL, USERNAME, PASSWORD.

Κλάση LoginForm:

Ο χρήστης (IT) για να συνδεθεί στη βάση χρειάζεται να δώσει το όνομα και τον κωδικό του. Η λειτουργία αυτή επιτελείται στην κλάση LoginForm και αν τα στοιχεία που συμπλήρωσε ο χρήστης επαληθεύονται, τότε ο (IT) γίνεται ενεργός δηλαδή αποκτά το πεδίο status την τιμή active και καλείται η επόμενη κλάση Menu. Αν τα στοιχεία δεν επαληθεύονται εμφανίζεται κατάλληλο μήνυμα.

Κλάση Menu:

Η κλάση αυτή μας δίνει όλες τις επιλογές που έχει ο υπεύθυνος πληροφορικής. Επιλέγοντας μία από αυτές και το κουμπί select οδηγούμαστε στην κλάση που έχουμε επιλέξει. Η Menu δίνει τις ακόλουθες επιλογές:

1. **INSERT/UPDATE/DELETE:** Ο IT με αυτή την επιλογή μπορεί να κάνει αλλαγή σε όποιον πίνακα επιλέξει στο επόμενο βήμα. Επιλέγοντας INSERT/UPDATE/DELETE εμφανίζεται το frame της κλάσης ChooseTable.
2. **SearchTrip:** (3.1.3.2) Εμφανίζεται το frame της κλάσης SearchTrip. Η κλάση αυτή υλοποιεί σε γραφικό περιβάλλον την procedure SearchTrip.
3. **Customer's Reservation:** (3.1.3.4) Εμφανίζεται το frame της κλάσης CustomerReservation. Η κλάση αυτή υλοποιεί σε γραφικό περιβάλλον την procedure customerReservation.
4. **Branch's Infos:** Εμφανίζεται το frame της κλάσης BranchInfo και βλέπει τα στοιχεία κάθε υποκαταστήματος, το ονοματεπώνυμο του διευθυντή του, το σύνολο κρατήσεων και το σύνολο εσόδων.
5. **Branch's Workers:** Εμφανίζεται το frame της κλάσης BranchWorker και βλέπει για κάθε υποκατάστημα το όνομα, επώνυμο και μισθό όλων των υπάλληλων και το συνολικό ποσό μισθών που πληρώνει.
6. **Show Log:** Εμφανίζεται το frame της κλάσης ShowLog και βλέπει όλες τις ενέργειες που έχουν καταγραφεί στον πίνακα log.
7. **Revenue and Cost:** (BONUS) Εμφανίζει το frame της κλάσης RevenueandCost και βλέπει σύνολο εσόδων και εξόδων ανά υποκατάστημα.

Κλάση ChooseTable:

Ο IT θα πρέπει να επιλέξει σε ποιόν από τους 16 πίνακες της βάσης θέλει να κάνει αλλαγές (insert/update/delete). Δεν του δίνεται η δυνατότητα να κάνει αλλαγές μόνο στον πίνακα log.

Κλάση Branch:

Σε αυτή την κλάση ο IT μπορεί να εισάγει/ενημερώσει/διαγράψει. Στο frame που εμφανίζεται ο χρήστης βλέπει ένα πίνακα με όλες τις εγγραφές που περιέχει ο πίνακας branch της βάσης. Μπορεί με τον cursor να επιλέξει μια από αυτές τις εγγραφές που υπάρχουν, τότε οι τιμές αυτής της εγγραφής θα συμπληρώσουν τα πεδία που υπάρχουν δίπλα. Μπορεί ο IT να αλλάξει τις τιμές των πεδίων ώστε να ενημερώσει την εγγραφή ή να μην τα αλλάξει και να επιλέξει διαγραφή. Τέλος του δίνεται η δυνατότητα να συμπληρώσει τα πεδία ώστε να εισάγει μια εγγραφή. Για κάθε ενέργεια εμφανίζει κατάλληλο μήνυμα ώστε να γνωρίζουμε αν η ενέργεια ολοκληρώθηκε. Αν δεν επιθυμεί να κάνει κάποια άλλη ενέργεια μπορεί να επιστρέψει στο Menu.

Στον κώδικα της κλάσης δημιουργούμε λίστα στην οποία αποθηκεύονται τα αποτελέσματα του query που κάνει select όλα τα πεδία του πίνακα.

Τα κουμπιά INSERT, DELETE, UPDATE εκτελούν το query που τους έχουμε καθορίσει (είναι αντίστοιχο της λειτουργίας που δηλώνει η ονομασία) και ενημερώνουμε τη λίστα με τα νέα δεδομένα.

Κλάσεις Worker,Admin,Manages,Phones,Driver,Guide,Language,Trip,Event,Destination,Travel_to,Reservation,IT,Offers,Reservation_offers:

Για κάθε πίνακα της βάσης εκτός του log έχουμε δημιουργήσει αντίστοιχη κλάση με όνομα αυτό του πίνακα της βάσης. Σε κάθε μία από αυτές τις κλάσεις εμφανίζεται frame που να υπακούει στα πεδία του πίνακα που αντιστοιχεί και δίνεται η δυνατότητα στον IT να εισάγει, ενημερώσει, διαγράψει. Ο κώδικας με τον οποίο υλοποιούνται αυτές οι κλάσεις είναι ίδιος με της κλάσης Branch, διαφοροποιείται μόνο στον αριθμό των πεδίων αφού κάθε πίνακας της βάσης έχει διαφορετικό πλήθος πεδίων.

Επίσης κάθε μία από αυτές τις κλάσεις έχει δική της κλάση SetterClassname (Classname το όνομα της κλάσης στην οποία αναφέρεται ο setter) με την οποία δημιουργούμε setter για να αποθηκεύουμε τις τιμές.

Κλάση SearchTrip:

Στο frame που εμφανίζεται πρέπει ο IT να εισάγει στα πεδία τον κωδικό της εταιρίας, ημερομηνία αναχώρησης και επιστροφής. Για αυτά τα στοιχεία που δίνει στους πίνακες δίπλα εμφανίζει τις ακόλουθες πληροφορίες:

- Κωδικός ταξιδιού που ανήκει σε αυτό το υποκατάστημα
- Κόστος ταξιδιού
- Μέγιστος αριθμός θέσεων
- Όνομα οδηγού ή ξεναγού
- Επώνυμο οδηγού ή ξεναγού
- Ημερομηνία αναχώρησης
- Ημερομηνία επιστροφής
- Αριθμό κρατήσεων
- Διαθέσιμες θέσεις

Για να εκτελέσουμε αυτή τη λειτουργία καλούμε την procedure SearchTrip (3.1.3.2) και τα αποτελέσματα τα αποθηκεύουμε σε δύο λίστες. Αν στις ημερομηνίες που δόθηκαν δεν υπάρχει ταξίδι από το υποκατάστημα που επιλέχθηκε εμφανίζεται μήνυμα και μας ενημερώνει.

Κλάση CustomerReservation:

Σε αυτή την κλάση καλούμε την procedure customerReservation (3.1.3.4) και με βάση το επώνυμο που δόθηκε στο πεδίο του frame εμφανίζονται στον πίνακα τα ονόματα και επώνυμα των πελατών με το επώνυμο αυτό και την προσφορά ταξιδιού στην οποία έχει γίνει εγγραφή. Πατώντας το select ο χρήστης εκτελείται το query για την procedure με το όρισμα που δόθηκε, τα αποτελέσματα αποθηκεύονται σε λίστα και τα εμφανίζει ο πίνακας του frame.

Κλάση BranchInfo:

Εκτελούμε κατάλληλο query και τα αποτελέσματα εμφανίζονται στον πίνακα του frame. Στον πίνακα βλέπουμε για κάθε υποκατάστημα τα στοιχεία του, το ονοματεπώνυμο του διευθυντή του, το σύνολο κρατήσεων και το σύνολο εσόδων.

Query που εκτελούμε:

Μέσα σε ένα εσωτερικό select δημιουργούμε προσωρινό πίνακα ο οποίος μετά από join με τους πίνακες branch, trip, reservation έχει δεδομένα για κάθε υποκατάστημα: τον κωδικό του κάθε ταξιδιού, το κόστος του κάθε ταξιδιού και το σύνολο κρατήσεων (κάνουμε group by tr_id). Για το τελικό αποτέλεσμα κάνουμε join μεταξύ του προσωρινού πίνακα που δημιουργήσαμε και των πινάκων manages για να πάρουμε τον διευθυντή του κάθε υποκαταστήματος και του worker για να πάρουμε το ονοματεπώνυμο του διευθυντή. Κάνοντας στο τέλος group by br_code παίρνουμε το πλήθος των κρατήσεων για κάθε υποκατάστημα και το κέρδος.

Κλάση BranchWorker:

Εκτελούμε κατάλληλα queries και τα αποτελέσματα εμφανίζονται στους πίνακες του frame. Στους πίνακες βλέπουμε για κάθε υποκατάστημα το όνομα, επώνυμο και μισθό όλων των υπάλληλων του και το συνολικό ποσό μισθών που πληρώνει.

Queries που εκτελούμε:

Στο πρώτο query που εκτελούμε επιλέγουμε με join από τους πίνακες branch και worker το κωδικό της εταιρίας και τα στοιχεία των υπαλλήλων της ονοματεπώνυμο και μισθό.

Στο δεύτερο query εμφανίζεται το συνολικό ποσό μισθών κάθε υποκαταστήματος αφού επιλέγουμε group by br_code.

Κλάση ShowLog:

Εκτελούμε κατάλληλο query και τα αποτελέσματα εμφανίζονται στον πίνακα του frame. Στον πίνακα βλέπουμε όλα τα περιεχόμενα του πίνακα της βάσης log.

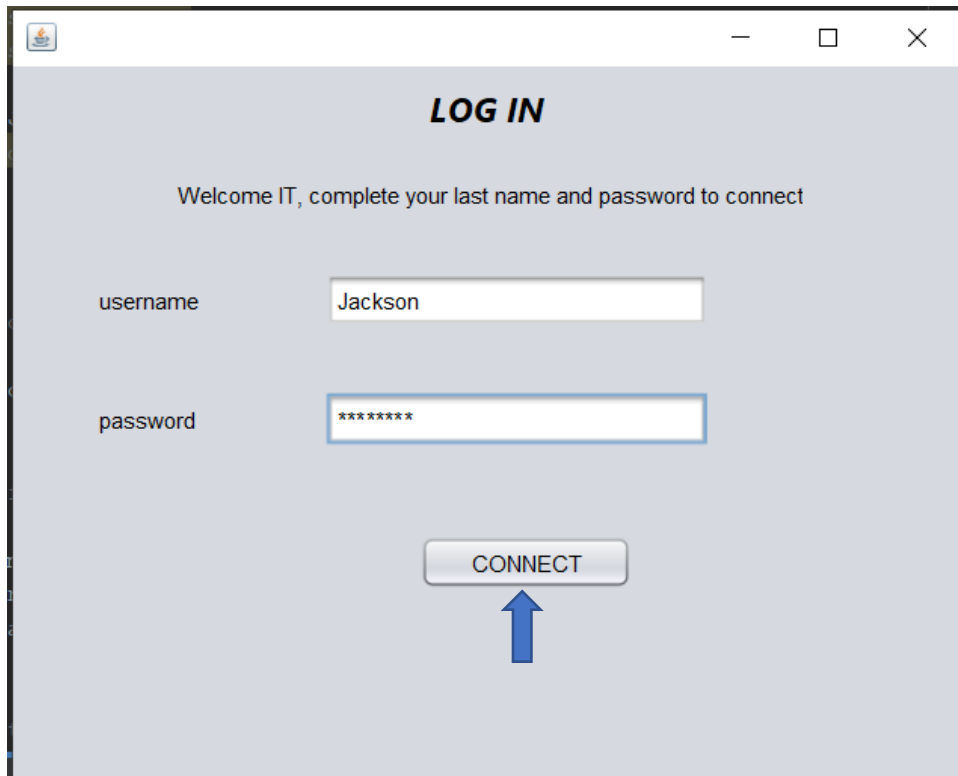
Query που εκτελούμε:

Κάνουμε select όλες τις στήλες του πίνακα log.

4. Σενάριο

Σενάριο: Θέλουμε να προσθέσουμε έναν υπάλληλο στο υποκατάστημα 2.

Κάνουμε log in με έναν IT.



The screenshot shows a web browser window with a light gray background. At the top, the title "LOG IN" is displayed in bold, italicized black font. Below the title, a message reads "Welcome IT, complete your last name and password to connect". There are two input fields: the first is labeled "username" and contains the text "Jackson"; the second is labeled "password" and contains seven asterisks "*****". Below the password field is a blue button labeled "CONNECT". A blue arrow points upwards towards the "CONNECT" button.

Επιλέγουμε “Branch’s Workers” και πατάμε “SELECT”.

MENU

Choose an option to continue

☐ INSERT/UPDATE/DELETE

☐ Search Trip

☐ Customer's Reservation

☐ Branch's Infos

☒ Branch's Workers

☐ Show Log

☐ Revenue and Cost

SELECT

LOG OUT

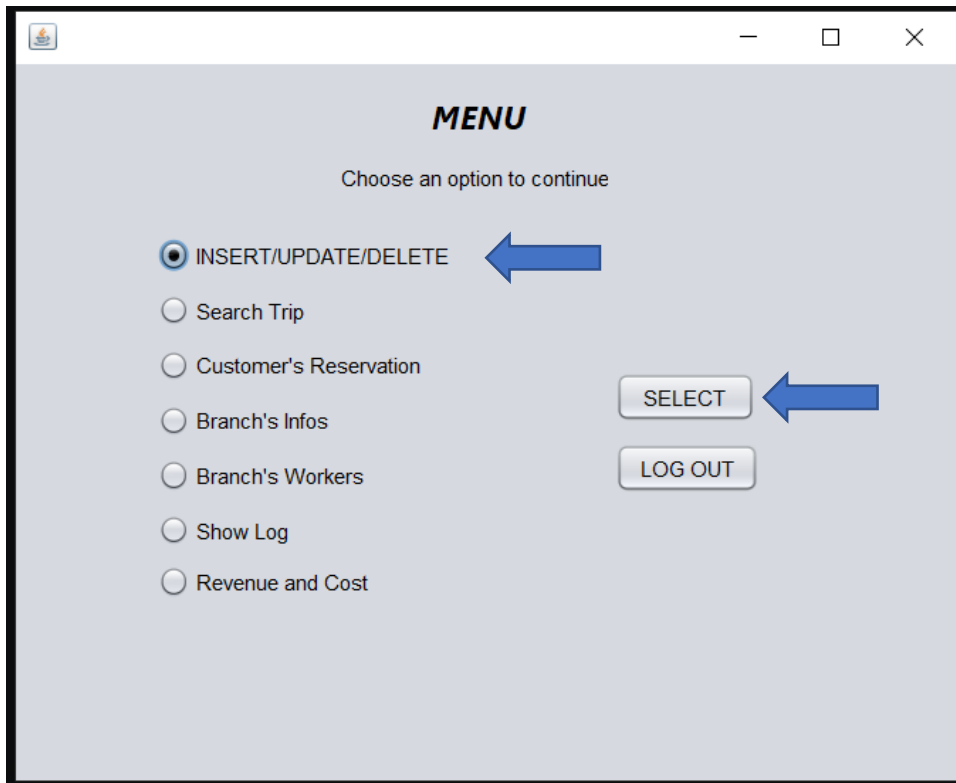
Βλέπουμε ότι το σύνολο των μισθών του υποκαταστήματος 2 είναι 14.000 και πατάμε “MENU” ώστε να γυρίσουμε στο μενού.

Menu

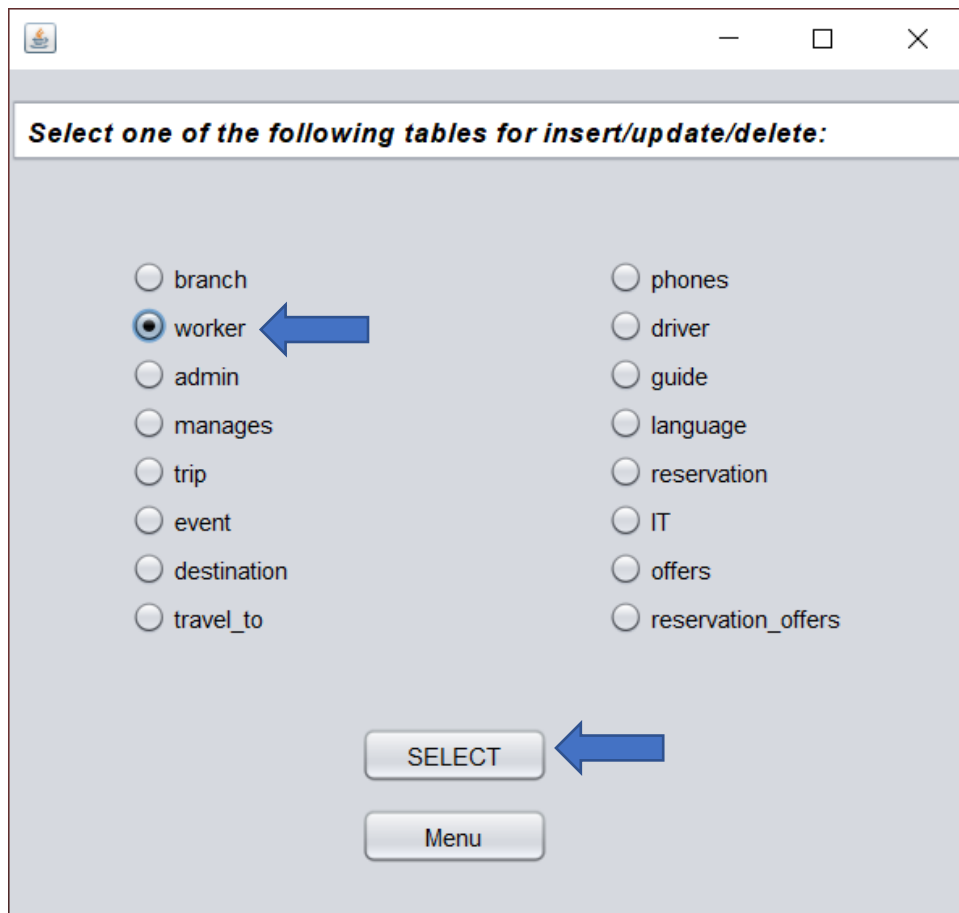
br_code	wrk_lname	wrk_name	wrk_salary
1	Martinez	Creath	1000.0
1	Martinez	Lachance	3000.0
1	Miller	Aarika	2000.0
1	Davis	Nadia	1000.0
1	Jackson	Nadiya	2000.0
2	Garcia	Nadiya	1000.0
2	Rodriguez	Etheline	4000.0
2	Anderson	Nadler	2000.0
2	Rodriguez	Aaren	4000.0
2	Johnson	Lachlan	3000.0
3	Johnson	Lachish	1000.0
3	Wilson	Aaron	2000.0
3	Williams	Nadler	4000.0
3	Jones	Creath	3000.0
3	Hernandez	Lachlan	1000.0

br_code	sumofsalary
1	9000.0
2	14000.0
3	11000.0
4	12000.0
5	13000.0
6	14000.0
7	7000.0
8	10000.0
9	15000.0
10	14000.0

Επιλέγουμε από το μενού την επιλογή “INSERT/UPDATE/DELETE” και πατάμε “SELECT”.



Επιλέγουμε τον πίνακα “worker” και πατάμε “SELECT”.



The screenshot shows a window titled "Select one of the following tables for insert/update/delete:". Inside the window, there are two columns of radio buttons, each followed by a table name. The first column contains: branch, worker, admin, manages, trip, event, destination, and travel_to. The second column contains: phones, driver, guide, language, reservation, IT, offers, and reservation_offers. The "worker" radio button is selected, and a blue arrow points to it. At the bottom of the window, there are two buttons: "SELECT" and "Menu". A blue arrow points to the "SELECT" button.

Select one of the following tables for insert/update/delete:

☐ branch

☒ worker

☐ admin

☐ manages

☐ trip

☐ event

☐ destination

☐ travel_to

☐ phones

☐ driver

☐ guide

☐ language

☐ reservation

☐ IT

☐ offers

☐ reservation_offers

SELECT

Menu

Βάζουμε στα πεδία τα στοιχεία του εργαζόμενου και πατάμε “INSERT”.

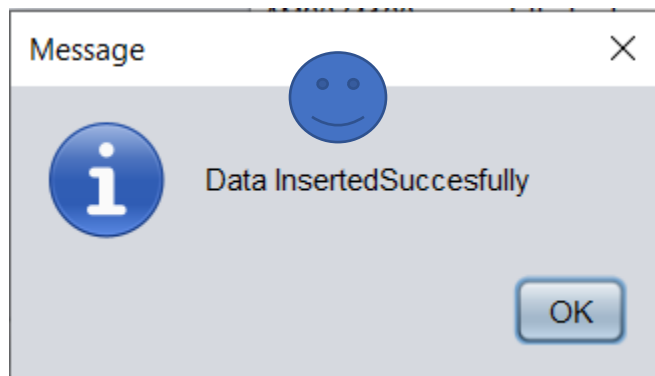
Form fields:

- wrk_AT: AM1231231
- wrk_name: Mitsos
- wrk_laname: Pitsos
- wrk_salary: 1234.0
- wrk_br_code: 2

Buttons: INSERT, UPDATE, DELETE, Menu

wrk_AT	wrk_name	wrk_laname	wrk_salary	wrk_br_code
AM1321755	Aarika	Garcia	5000.0	10
AM1544854	Lachish	Thompson	3000.0	8
AM1619922	Etheline	Rodriguez	4000.0	2
AM1858528	Creath	Martinez	1000.0	1
AM1893557	Nadiya	Davis	1000.0	9
AM2129858	Creigh	Williams	2000.0	7
AM2512172	Ethelred	Thompson	4000.0	4
AM2674183	Ethelred	Martinez	4000.0	10
AM2761138	Lach	Moore	1000.0	4
AM2844976	Lachish	Williams	4000.0	4
AM2885413	Nadler	Thompson	2000.0	5
AM3128853	Lachance	Anderson	2000.0	8
AM3132639	Nadler	Williams	4000.0	3
AM3656761	Aaren	Wilson	4000.0	5
AM3982193	Aaron	Miller	1000.0	7
AM4171974	Nadiya	Jackson	2000.0	1
AM4194673	Ethelyn	Rodriguez	4000.0	5
AM4832485	Lachlan	Anderson	4000.0	6
AM4843271	Creigh	Martinez	4000.0	6
AM4883967	Nadine	Thompson	1000.0	7
AM5188847	Etheline	Smith	2000.0	6
AM5231946	Aarika	Taylor	4000.0	6
AM5443826	Creath	Jones	3000.0	3
AM5697977	Ethelyn	Thompson	3000.0	5
AM5886366	Nadiya	Garcia	1000.0	2

Βγαίνει μήνυμα επιτυχημένης εισαγωγής.



Πατάμε “MENU” και επιστρέφουμε στο μενού.

The screenshot shows a database application window. On the left, there is a form with input fields for 'wrk_AT', 'wrk_name', 'wrk_lanme', 'wrk_salary', and 'wrk_br_code'. Below these fields are buttons for 'INSERT', 'UPDATE', and 'DELETE'. A 'Menu' button is located below these, with a blue arrow pointing to it. On the right, there is a table with the following data:

wrk_AT	wrk_name	wrk_lname	wrk_salary	wrk_br_code
AM1231231	Mitsos	Pitsos	1234.0	2
AM1321755	Aarika	Garcia	5000.0	10
AM1544854	Lachish	Thompson	3000.0	8
AM1619922	Etheline	Rodriguez	4000.0	2
AM1858528	Creath	Martinez	1000.0	1
AM1893557	Nadiya	Davis	1000.0	9
AM2129858	Creigh	Williams	2000.0	7
AM2512172	Ethelred	Thompson	4000.0	4
AM2674183	Ethelred	Martinez	4000.0	10
AM2761138	Lach	Moore	1000.0	4
AM2844976	Lachish	Williams	4000.0	4
AM2885413	Nadler	Thompson	2000.0	5
AM3128853	Lachance	Anderson	2000.0	8
AM3132639	Nadler	Williams	4000.0	3
AM3656761	Aaren	Wilson	4000.0	5
AM3982193	Aaron	Miller	1000.0	7
AM4171974	Nadiya	Jackson	2000.0	1
AM4194673	Ethelyn	Rodriguez	4000.0	5
AM4832485	Lachlan	Anderson	4000.0	6
AM4843271	Creigh	Martinez	4000.0	6
AM4883967	Nadine	Thompson	1000.0	7
AM5188847	Etheline	Smith	2000.0	6
AM5231946	Aarika	Taylor	4000.0	6
AM5443826	Creath	Jones	3000.0	3
AM5697977	Ethelyn	Thompson	3000.0	5

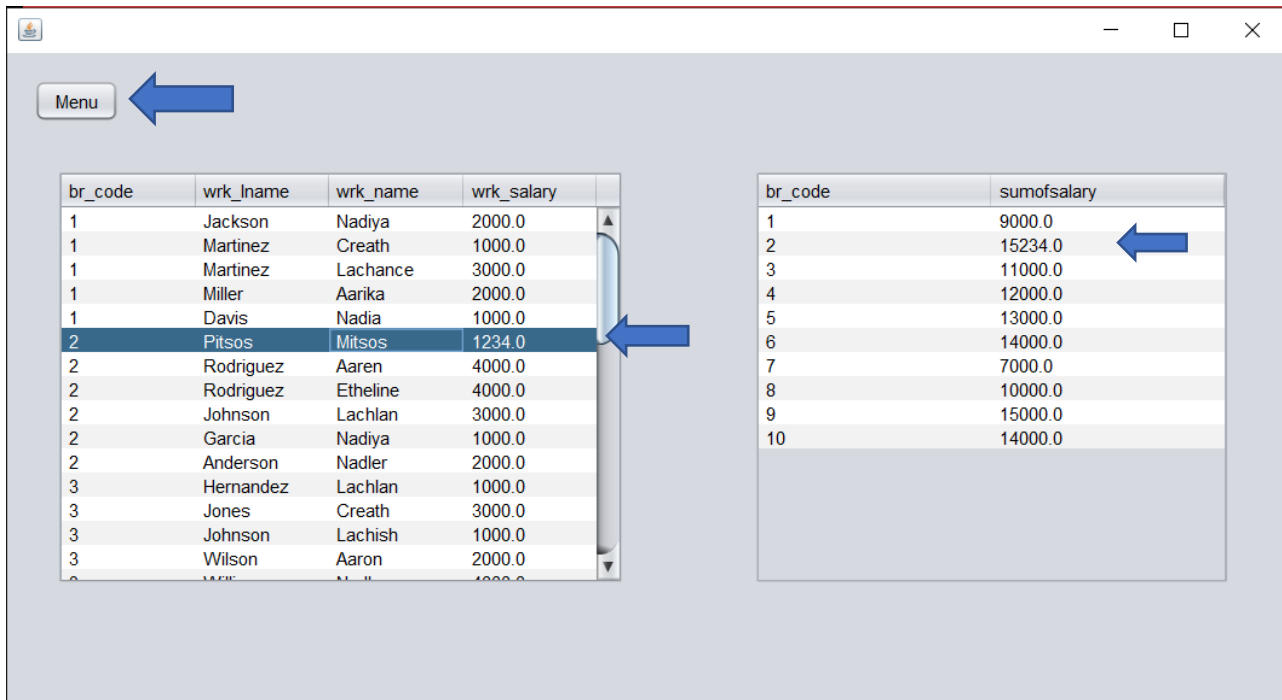
Επιλέγουμε “Branch’s Workers” και πατάμε “SELECT”.

The screenshot shows a menu screen titled "MENU". Below the title, it says "Choose an option to continue". There is a list of options with radio buttons next to them:

- ☐ INSERT/UPDATE/DELETE
- ☐ Search Trip
- ☐ Customer's Reservation
- ☐ Branch's Infos
- ☒ Branch's Workers
- ☐ Show Log
- ☐ Revenue and Cost

On the right side of the screen, there are two buttons: "SELECT" and "LOG OUT". A blue arrow points to the "SELECT" button.

Βλέπουμε ότι στο υποκατάστημα με br_code=2 το άθροισμα των μιστών τελειώνει σε 234 και άρα επηρεάστηκε από την αλλαγή που κάναμε.

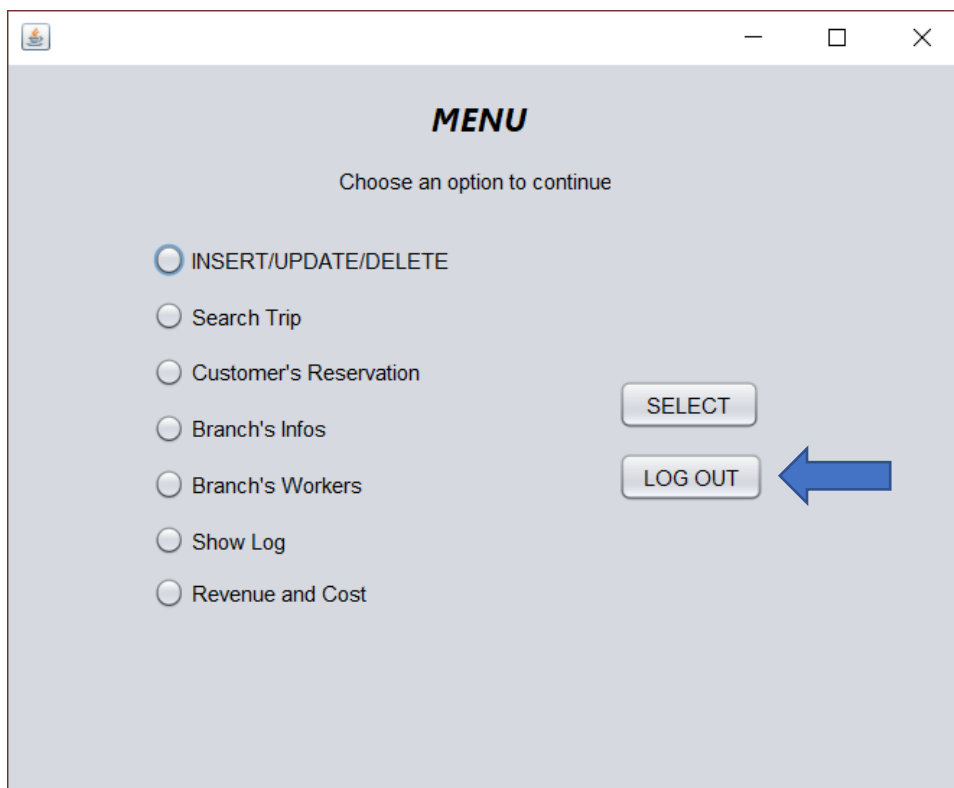


Menu

br_code	wrk_lname	wrk_name	wrk_salary
1	Jackson	Nadiya	2000.0
1	Martinez	Creath	1000.0
1	Martinez	Lachance	3000.0
1	Miller	Aarika	2000.0
1	Davis	Nadia	1000.0
2	Pitsos	Mitsos	1234.0
2	Rodriguez	Aaren	4000.0
2	Rodriguez	Etheline	4000.0
2	Johnson	Lachlan	3000.0
2	Garcia	Nadiya	1000.0
2	Anderson	Nadler	2000.0
3	Hernandez	Lachlan	1000.0
3	Jones	Creath	3000.0
3	Johnson	Lachish	1000.0
3	Wilson	Aaron	2000.0

br_code	sumofsalary
1	9000.0
2	15234.0
3	11000.0
4	12000.0
5	13000.0
6	14000.0
7	7000.0
8	10000.0
9	15000.0
10	14000.0

Πατάμε “MENU”.



MENU

Choose an option to continue

☐ INSERT/UPDATE/DELETE

☐ Search Trip

☐ Customer's Reservation

☐ Branch's Infos

☐ Branch's Workers

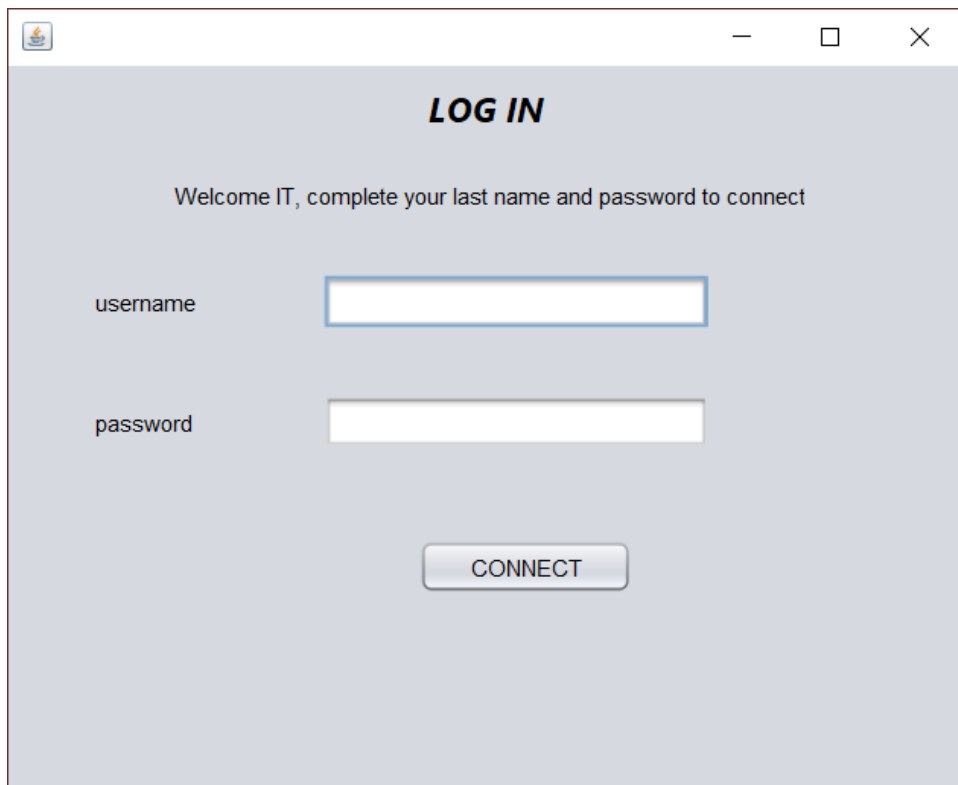
☐ Show Log

☐ Revenue and Cost

SELECT

LOG OUT

Κάνουμε log out πατώντας στο “LOG OUT”.



LOG IN

Welcome IT, complete your last name and password to connect

username

password

CONNECT

5. Extra λειτουργία

Ερώτημα 3.2.3

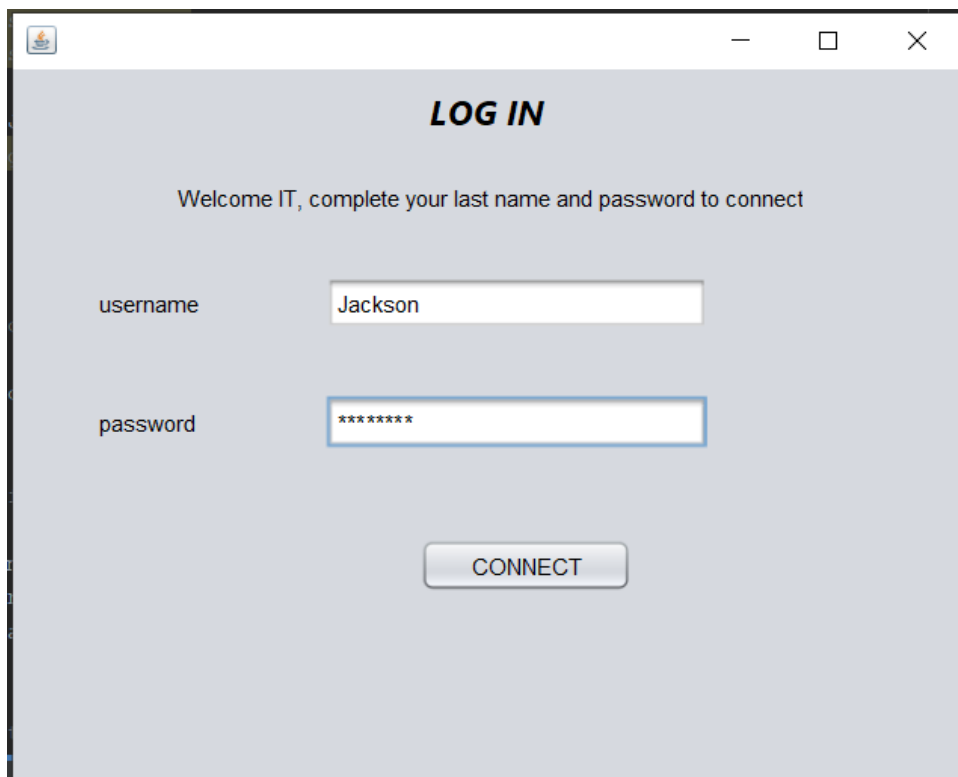
Κλάση RevenueAndCost:

Εκτελούμε κατάλληλο query και τα αποτελέσματα εμφανίζονται στον πίνακα του frame. Στον πίνακα βλέπουμε για όλα τα υποκαταστήματα το συνολικό ποσό εξόδων δηλαδή το άθροισμα των μισθών των υπαλλήλων και το συνολικό ποσό εσόδων δηλαδή κρατήσεις επί κόστος ταξιδιού.

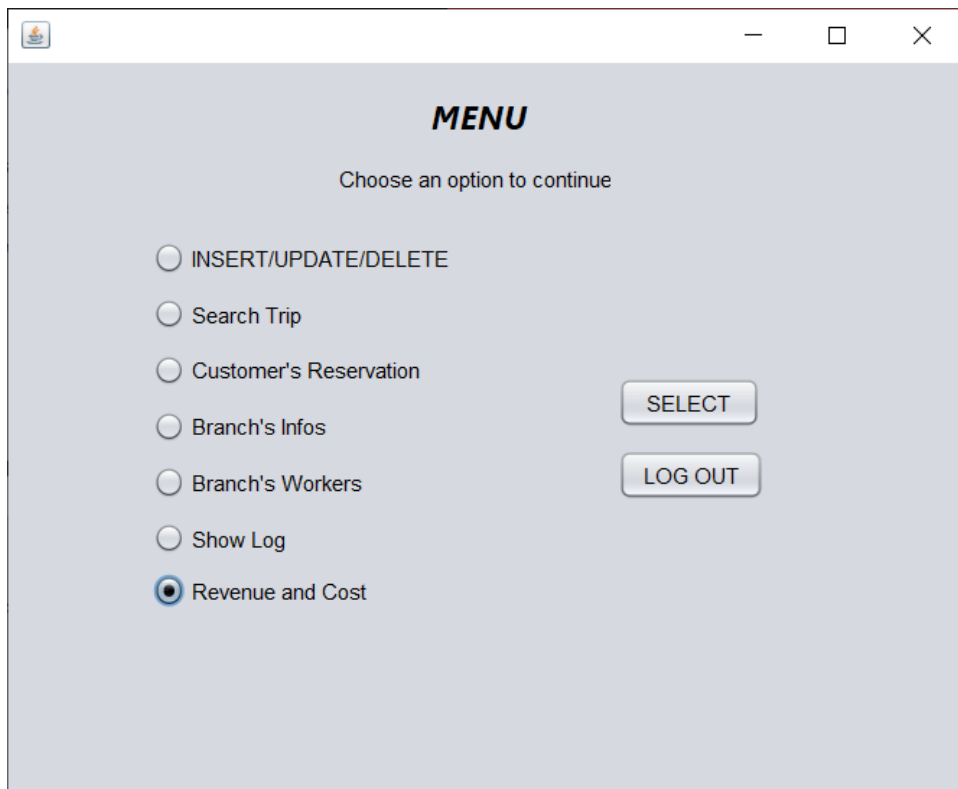
Query που εκτελούμε:

Για να εμφανίσουμε το συνολικό ποσό εξόδων και εξόδων συνδυάζουμε τα queries των κλάσεων BranchInfo και BranchWorker. Για το συνολικό ποσό εξόδων δημιουργούμε προσωρινό πίνακα με τις συνολικές κρατήσεις ανά ταξίδι και το κόστος του ταξιδιού. Το join αυτού του πίνακα με τον άλλο προσωρινό πίνακα που υπολογίζει το σύνολο των εξόδων ανά υποκατάστημα μας δίνει τον τελικό πίνακα που είναι το ζητούμενο.

Κάνουμε log in με έναν IT.



Επιλέγουμε “Revenue and Cost” και πατάμε “SELECT”.



MENU

Choose an option to continue

☐ INSERT/UPDATE/DELETE

☐ Search Trip

☐ Customer's Reservation

☐ Branch's Infos

☐ Branch's Workers

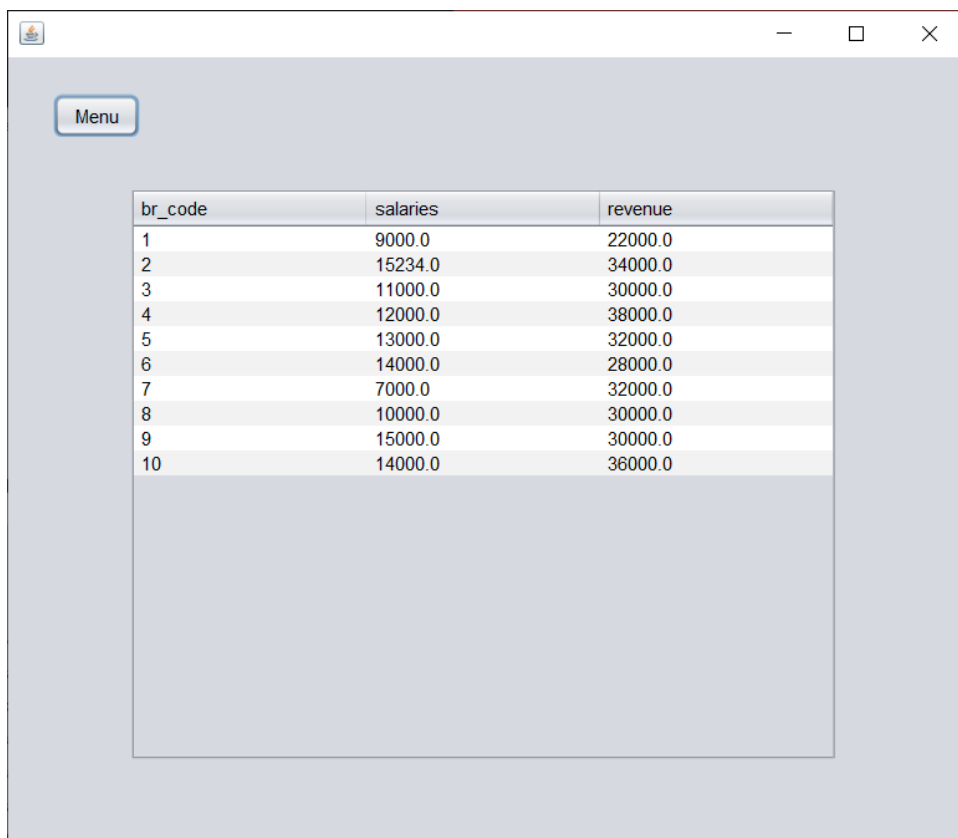
☐ Show Log

☒ Revenue and Cost

SELECT

LOG OUT

Εμφανίζεται στην οθόνη το αποτέλεσμα της εκτέλεσης του query: κωδικός εταιρίας, έξοδα, έσοδα



br_code	salaries	revenue
1	9000.0	22000.0
2	15234.0	34000.0
3	11000.0	30000.0
4	12000.0	38000.0
5	13000.0	32000.0
6	14000.0	28000.0
7	7000.0	32000.0
8	10000.0	30000.0
9	15000.0	30000.0
10	14000.0	36000.0