**Κεφάλαιο 1**

1. Φασδαδς
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.

Table

Description automatically generated**Παράδειγμα Υλοποίησης:**

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

Text

Description automatically generated

Έπειτα ας δούμε τους πίνακες worker και driver:

Text

Description automatically generated(O 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):

Table

Description automatically generated

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 συνάρτηση:

Table

Description automatically generatedTable

Description automatically generatedΑς συγκρίνουμε τους πίνακες τώρα με τους αντίστοιχους πίνακες πριν:

Graphical user interface, text, application

Description automatically generated

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 συνάρτηση 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),

        wrk\_name\_ CHAR(20),

        wrk\_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 AS Trip\_Cost, tr\_maxseats AS MaxSeats, reservations,

                seatdiff AS Available\_Seats, a.wrk\_name AS Driver\_Name, a.wrk\_lname AS Driver\_LName,

                b.wrk\_name AS Guide\_Name, b.wrk\_lname AS Guide\_LName, tr\_departure AS Departure, tr\_return AS Returning

                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:

A picture containing table

Description automatically generated

Graphical user interface, application, Teams

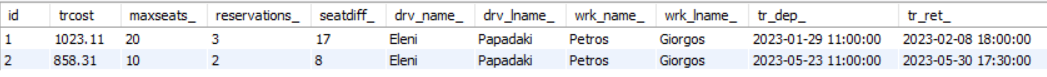
Description automatically generated

Καλούμε την συνάρτηση:



Από τον παραπάνω πίνακα παρατηρούμε πως μόνο ένα ταξίδι καλύπτει τις προϋποθέσεις των ορισμάτων της συνάρτησης, και αυτό είναι και αυτό που εμφανίζει και η συνάρτηση.

Καλούμε ξανά την συνάρτηση με άλλα ορίσματα για να σιγουρευτούμε ότι δουλεύει για για περισσότερα από 1 ταξίδια:



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