2024-2025

ΕΡΓΑΣΤΗΡΙΟ ΒΑΣΕΩΝ ΔΕΔΟΜΕΝΩΝ

Project

Ζεκυριάς Απόστολος (1100554)

Καραμαλίκης Ανδρέας (1100562)

**Γενικές Πληροφορίες**

Στις επόμενες σελίδες παρουσιάζονται οι απαντήσεις της ομάδας μας στο Project του εργαστηρίου. Σε αυτήν τη σελίδα παρέχονται πληροφορίες σχετικά με τα μέλη της ομάδας.

Η ομάδα αποτελέιται από τους εξής φοιτητές:

Ζεκυριάς Απόστολος

Καραμαλίκης Ανδρέας

**Αναλυτικότερες Πληροφορίες:**

Ανδρέας Καραμαλίκης 1100562

[up1100562@ac.upatras.gr](mailto:up1100613@ac.upatras.gr)

Φοιτητής 3ου έτους

Απόστολος Ζεκυριάς 1100554

[up1100554@ac.upatras.gr](mailto:up1100554@ac.upatras.gr)

Φοιτητής 3ου έτους

**Περιεχόμενα**

**Μέρος Α: Σχεδιασμός ΒΔ και SQL**

[Κεφάλαιο 1: Δημιουργία της ΒΔ 3](#_Toc187876128)

[Σχεσιακό Διάγραμμα Της Συνολικής Αναθεωρημένης ΒΔ 3](#_Toc187876129)

[Περιγραφή Παραδοχών Σχεδίασης 4](#_Toc187876130)

[Νέοι Πίνακες 4](#_Toc187876131)

[Πίνακας Συναυλιών (Concert) 4](#_Toc187876132)

[Πίνακας Χώρων Διεξαγωγής (Venues) 5](#_Toc187876133)

[Πίνακας Ιστορικού Συναυλιών (ConcertHistory) 5](#_Toc187876134)

[Πίνακας Διαχειριστών Βάσης Δεδομένων (DBA) 5](#_Toc187876135)

[Πίνακας Καταγραφής Ενεργειών (Log) 6](#_Toc187876136)

[Εντολές SQL Για Τη Δημιουργία Της ΒΔ 6](#_Toc187876137)

[Κεφάλαιο 2: Δημιουργία Stored Procedures 6](#_Toc187876138)

[Stored Procedure CalculateVenueScore (3.1.3.1) 6](#_Toc187876139)

[Κώδικας SQl 6](#_Toc187876140)

[Screenshots Από Την Εκέλεση 7](#_Toc187876141)

# **Κεφάλαιο 1: Δημιουργία της ΒΔ**

## A computer screen shot of a diagram Description automatically generatedΣχεσιακό Διάγραμμα Της Συνολικής Αναθεωρημένης ΒΔ

## Περιγραφή Παραδοχών Σχεδίασης

Κατά τη διαδικασία σχεδίασης της βάσης δεδομένων, βασιστήκαμε στις ακόλουθες παραδοχές:

1. Κάθε συναυλία αντιστοιχεί σε έναν καλλιτέχνη και έναν χώρο διεξαγωγής.
2. Ο πίνακας του ιστορικού συναυλιών (ConcertHistory ) δεν απαιτεί περιορισμούς αναφορικής ακεραιότητας με άλλους πίνακες.
3. Για λόγους διευκόλυνσης, αποφασίσαμε ότι το πεδίο ReqCapacity του πίνακα Concert και το πεδίο NumTickets του πίνακα ConcertHistory θα έχουν τα ίδια δεδομένα.

## Νέοι Πίνακες

### Πίνακας Συναυλιών (Concert)

Ο πίνακας Concert αποθηκεύει πληροφορίες για τις συναυλίες. Περιέχει τα ακόλουθα πεδία:

* ConID: Μοναδικό αναγνωριστικό της συναυλίας.
* VenID: Μοναδικό αναγνωριστικό του χώρου διεξαγωγής.
* ArtistID: Μοναδικό αναγνωριστικό του καλλιτέχνη.
* ConDate: Ημερομηνία διεξαγωγής της συναυλίας.
* Status: Κατάσταση της συναυλίας (π.χ. scheduled, completed, cancelled).
* ReqCapacity: Η απαιτούμενη χωρητικότητα για να πραγματοποιηθεί η συναυλία.

### Πίνακας Χώρων Διεξαγωγής (Venues)

Ο πίνακας Venues περιέχει στοιχεία για τους διαθέσιμους χώρους διεξαγωγής. Περιέχει τα ακόλουθα πεδία:

* VenID: Μοναδικό αναγνωριστικό του χώρου.
* Name: Το όνομα του χώρου.
* Location: Η τοποθεσία του χώρου.
* Capacity: Η χωρητικότητα του χώρου.
* ConcertsHeld: Ο συνολικός αριθμός των συναυλιών που έχουν πραγματοποιηθεί στον συγκεκριμένο χώρο.
* DateOpened: Η ημερομηνία κατά την οποία άνοιξε ο χώρος διεξαγωγής.

### Πίνακας Ιστορικού Συναυλιών (ConcertHistory)

Ο πίνακας ConcertHistory αποθηκεύει τις ολοκληρωμένες ή ακυρωμένες συναυλίες. Περιέχει τα ακόλουθα πεδία:

* ConcertID: Μοναδικό αναγνωριστικό της συναυλίας.
* ArtistID: Μοναδικό αναγνωριστικό του καλλιτέχνη.
* VenueId: Μοναδικό αναγνωριστικό του χώρου διεξαγωγής.
* NumTickets: Ο αριθμός των εισιτηρίων που πωλήθηκαν.
* ConcertDate: Ημερομηνία διεξαγωγής (ή NULL για ακυρωμένες συναυλίες).
* Status: Κατάσταση της συναυλίας (completed, cancelled).

### Πίνακας Διαχειριστών Βάσης Δεδομένων (DBA)

Ο πίνακας DBA αποθηκεύει πληροφορίες για τους διαχειριστές βάσεων δεδομένων. Περιέχει τα ακόλουθα πεδία:

* Username: Το όνομα χρήστη κάθε διαχειριστή (το οποίο αποτελεί και μοναδικό αναγνωριστικό).
* start\_date: Ημερομηνία ανάληψης καθηκόντων (δεν μπορεί να είναι NULL).
* end\_date: Ημερομηνία λήξης καθηκόντων (NULL εάν είναι ενεργός).

### Πίνακας Καταγραφής Ενεργειών (Log)

Ο πίνακας Log καταγράφει τις ενέργειες των διαχειριστών.

* Username: Αναφορά στον διαχειριστή που πραγματοποίησε την ενέργεια.
* Time: Η ημερομηνία και η ώρα της ενέργειας.
* Αction: Περιγραφή της ενέργειας (π.χ. Insert, Update, Delete).
* Table: Το όνομα του πίνακα που επηρεάστηκε (π.χ. person, band, album, concert, venue).

## Εντολές SQL Για Τη Δημιουργία Της ΒΔ

# **Κεφάλαιο 2: Δημιουργία Stored Procedures**

## Stored Procedure CalculateVenueScore (3.1.3.1)

Η διαδικασία CalculateVenueScore υπολογίζει μια βαθμολογία για έναν χώρο διεξαγωγής με βάση τη χωρητικότητα, τον αριθμό συναυλιών που έχουν πραγματοποιηθεί και τα χρόνια λειτουργίας. Ανακτά τα δεδομένα του χώρου από τον πίνακα venues, υπολογίζει επιμέρους βαθμολογίες (χωρητικότητας, συναυλιών και χρόνων λειτουργίας) και επιστρέφει το συνολικό σκορ. Οι βαθμολογίες βασίζονται σε στρογγυλοποιήσεις και συγκεκριμένους συντελεστές για κάθε παράμετρο. Η διαδικασία αυτή αποτελεί έναν αρκετά τυποποιημένο τρόπο αξιολόγησεις.

### Κώδικας SQl

DELIMITER $

CREATE PROCEDURE CalculateVenueScore(IN venueId INT, OUT venueScore INT)

BEGIN

    DECLARE capacityScore INT;

    DECLARE concertsScore INT;

    DECLARE yearsScore INT;

    DECLARE yearsOfOperation INT;

    SELECT

        Capacity,

        ConcertsHeld,

        TIMESTAMPDIFF(YEAR, DateOpened, CURDATE()) AS YearsOfOperation

    INTO

        @venueCapacity,

        @venueConcerts,

        yearsOfOperation

    FROM

        venues

    WHERE

        VenId = venueId;

--Το FLOOR στρογγυλοποιεί το αποτέλεσμα προς τα κάτω. Το χρησιμοποιούμε ώστε να μην έχουμε δεκαδικόυς αριθμόυς

    SET capacityScore = FLOOR(@venueCapacity / 1000) \* 1;

    SET concertsScore = FLOOR(@venueConcerts / 100) \* 3;

    SET yearsScore = yearsOfOperation \* 2;

    SET venueScore = capacityScore + concertsScore + yearsScore;

END $

DELIMITER ;

Κώδικας Εκτέλεσης Του Procedure

CALL CalculateVenueScore(1, @score);

SELECT @score AS VenueScore;

### Screenshots Από Την Εκέλεση

A screen shot of a computer

Description automatically generated

Αυτό το αποτέλεσμα προκύπτει για το Venue με τα ακόλουθα δεδομένα:

(1, 'Athens Concert Hall', 'Athens, Greece', 2000, 350, '2005-06-15')

## Stored Procedure ManageConcerts (3.1.3.2)

Η διαδικασία ManageConcerts διαχειρίζεται συναυλίες για έναν συγκεκριμένο καλλιτέχνη και ημερομηνία με βάση την ενέργεια που της ζητείται. Ελέγχει αν υπάρχει ήδη συναυλία για τον καλλιτέχνη και την ημερομηνία, και ανάλογα με το action εκτελεί εισαγωγή νέας συναυλίας, ακύρωση υπάρχουσας, ή επανενεργοποίηση ακυρωμένης. Για κάθε περίπτωση, επιστρέφει ένα κατάλληλο μήνυμα που περιγράφει το αποτέλεσμα της ενέργειας.

### Κώδικας SQl

DELIMITER $

CREATE PROCEDURE ManageConcert(

    IN artistId INT,

    IN concertDate DATE,

    IN action CHAR(1)

)

BEGIN

    DECLARE existingStatus ENUM('Scheduled', 'Cancelled', 'Completed');

    DECLARE message VARCHAR(255);

    -- Elegxei an yparxei synaulia gia ton kallitexnh kai thn hmeromhnia

    SELECT Status

    INTO existingStatus

    FROM concert

    WHERE ArtistId = artistId AND ConDate = concertDate;

    -- Cases gia to action

    CASE action

        WHEN 'i' THEN

            -- An yparxei

            IF existingStatus IS NOT NULL THEN

                IF existingStatus = 'Scheduled' THEN

                   SET message = 'A concert is already scheduled on this date.';

                ELSEIF existingStatus = 'Cancelled' THEN

                   SET message = 'A cocnert is cancelled on this date.';

                ELSE

                    SET message = 'The concert has already been completed.';

                END IF;

            ELSE

                -- Kainourgia synaulia me status scheduled

                INSERT INTO concert (VenId, ArtistId, ConDate, Status, ReqCapacity)

                VALUES (1, artistId, concertDate, 'Scheduled', 1000); -- Gia eukolia ebala oti oles oi kainourges synaulies programmatizontai sto venue 1 kai me reqcapacity 1000

                SET message = 'A new concert was scheduled successfully.';

            END IF;

        WHEN 'c' THEN

            -- Akyrwsh

            IF existingStatus IS NULL THEN

                SET message = 'No concert exists on this date.';

            ELSEIF existingStatus = 'Cancelled' THEN

               SET message = 'The concert is already cancelled.';

            ELSE

                UPDATE concert

                SET Status = 'Cancelled'

                WHERE ArtistId = artistId AND ConDate = concertDate;

                SET message = 'The concert was cancelled successfully.';

            END IF;

        WHEN 'a' THEN

            -- Ksana energopoiei mia akyrwmenh synaulia

            IF existingStatus IS NULL THEN

                SET message = 'No concert exists on this date.';

            ELSEIF existingStatus = 'Scheduled' THEN

                SET message = 'A concert is already scheduled on this date.';

            ELSEIF existingStatus = 'Cancelled' THEN

                UPDATE concert

                SET Status = 'Scheduled'

                WHERE ArtistId = artistId AND ConDate = concertDate;

                SET message = 'The concert was rescheduled successfully.';

            END IF;

        ELSE

        -- Gia opoiondhpote allo xarakthra

            SET message = 'I dont know what to do with that bro.';

    END CASE;

      -- Epistrefei to mhnuma sto telos

    SELECT message AS ResultMessage;

END$

### Screenshots Από Την Εκέλεση

### Αποτελέσματα αν ο χρήστης εισάγει i

A screen shot of a program

Description automatically generated

A screen shot of a computer program

Description automatically generated

A screen shot of a computer program

Description automatically generated

A screen shot of a program

Description automatically generated

### Αποτελέσματα αν ο χρήστης εισάγει c

A computer screen with white text

Description automatically generated

A screen shot of a computer program

Description automatically generated

### Αποτελέσματα αν ο χρήστης εισάγει a

A screen shot of a computer program

Description automatically generated

A screen shot of a concert program

Description automatically generated

## Stored Procedure (3.1.3.3)