



**ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ ΣΧΟΛΗ  
ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ ΚΑΙ ΜΗΧΑΝΙΚΩΝ  
ΥΠΟΛΟΓΙΣΤΩΝ ΕΡΓΑΣΤΗΡΙΟ ΣΥΣΤΗΜΑΤΩΝ ΒΑΣΕΩΝ  
ΓΝΩΣΕΩΝ ΚΑΙ ΔΕΔΟΜΕΝΩΝ Ακ. έτος 2024-2025, 6ο εξάμηνο,  
ΣΗΜΜΥ**

Σχεδίαση και Υλοποίηση Συστήματος Διαχείρισης για το Μουσικό  
Φεστιβάλ Pulse University

**Μάθημα:** Βάσεις Δεδομένων

**Διδάσκοντες:** Δημήτριος Τσουμάκος, Μάριος Κόνιαρης

**Φοιτήτριες:** Ελένη Στυλιανού-ge21708, Χλόη Χρίστου-ge21702

**Ομάδα:** 8

**Σχολή:** ΣΕΜΦΕ

**Εξάμηνο:** 8<sup>ο</sup>

**Emails:** [elenistylianou03@live.com](mailto:elenistylianou03@live.com), [chloechristou2003@gmail.com](mailto:chloechristou2003@gmail.com)

**GitHub Link:** [https://github.com/stylianoueleni/db\\_assignment](https://github.com/stylianoueleni/db_assignment)

## Περίληψη

Η εργασία επικεντρώνεται στη σχεδίαση και υλοποίηση σχεσιακής βάσης δεδομένων για το διεθνές μουσικό φεστιβάλ *Pulse University*, με σκοπό την αποδοτική διαχείριση της διοργάνωσης και της λειτουργίας του. Το έργο περιλαμβάνει τον πλήρη σχεδιασμό ER διαγράμματος, τη μετατροπή του σε σχεσιακό μοντέλο με ενσωμάτωση περιορισμών ακεραιότητας, την ανάπτυξη της ΒΔ με SQL, καθώς και την υλοποίηση ερωτημάτων, triggers και ευρετήριων που ανταποκρίνονται στις ανάγκες του σεναρίου. Η ορθότητα της λύσης επαληθεύεται με ενδεικτικά δεδομένα, διασφαλίζοντας την επιστροφή έγκυρων αποτελεσμάτων για όλα τα ζητούμενα.

## Περιεχόμενα

- Εισαγωγή
- Περιγραφή Φεστιβάλ

## ΜΕΡΟΣ Α

- ER Διάγραμμα
- Σχεσιακό Μοντέλο
- Υλοποίηση της Βάσης Δεδομένων σε SQL
  1. Δημιουργία της Βάσης
  2. Πίνακες Αναφοράς (Lookup Tables)
  3. Πίνακες Οντοτήτων
  4. Triggers
  5. Stored Procedures
  6. Views (Οψεις)
  7. Ευρετήρια (Indexes)
  8. Υποσυστήματα και Υποστηρικτικοί Πίνακες
  9. Σχολιασμός Εισαγωγής Δεδομένων

## ΜΕΡΟΣ Β

- Αναφορά Εφαρμογής και Υλοποίησης Ερωτημάτων Βάσης Δεδομένων Μουσικού Φεστιβάλ
  1. Αρχιτεκτονική Εφαρμογής και Τεχνική Υλοποίηση
  2. Υλοποίηση και Βελτιστοποίηση Ερωτημάτων
  3. Διεπαφή Χρήστη και Δυνατότητες Οπτικοποίησης
  4. Ανάλυση Απόδοσης και Τεχνικές Γνώσεις
  5. Προκλήσεις και Λύσεις
  6. Ειδική Εστίαση στις Υλοποιήσεις Ερωτημάτων
  7. Σύνοψη Τεχνικών Βελτιστοποίησης
  8. Περίληψη Πλαισίου Δοκιμών
  9. Εντολές εύρεσης δεδομένων για εισαγωγή στα ερωτήματα από τον χρήστη

## Εισαγωγή

Η εργασία αυτή εκπονήθηκε στο πλαίσιο του μαθήματος «Βάσεις Δεδομένων» του Τμήματος Ηλεκτρολόγων Μηχανικών και Μηχανικών Υπολογιστών (ΣΗΜΜΥ) του ΕΜΠ και αφορά τη σχεδίαση και υλοποίηση ενός ολοκληρωμένου συστήματος βάσης δεδομένων για τη διοργάνωση του μουσικού φεστιβάλ Pulse University.

Το φεστιβάλ διεξάγεται ετησίως σε διαφορετικές τοποθεσίες ανά τον κόσμο και χαρακτηρίζεται από σύνθετη δομή, καθώς περιλαμβάνει καλλιτεχνικά δρώμενα, τεχνική και οργανωτική υποστήριξη, αγοραπωλησία εισιτηρίων και δυνατότητα αξιολόγησης από το κοινό.

Η βάση δεδομένων που αναπτύχθηκε αποσκοπεί στην αξιόπιστη, αποδοτική και ασφαλή αποθήκευση και ανάκτηση δεδομένων που σχετίζονται με κάθε πτυχή του φεστιβάλ. Ειδική έμφαση δόθηκε στη διατήρηση της ακεραιότητας των δεδομένων, στην υποστήριξη σύνθετων ερωτημάτων και στην εξασφάλιση καλής απόδοσης.

Η αναφορά που ακολουθεί περιγράφει τη μεθοδολογία σχεδίασης, την υλοποίηση της βάσης δεδομένων, καθώς και τις τεχνικές επιλογές που υιοθετήθηκαν, αξιολογώντας την αποτελεσματικότητα του τελικού συστήματος σε σχέση με τις απαιτήσεις του σεναρίου.

## Περιγραφή Φεστιβάλ

Το *Pulse University* αποτελεί ένα διεθνές μουσικό φεστιβάλ που διεξάγεται σε ετήσια βάση, κάθε φορά σε διαφορετική τοποθεσία ανά τον κόσμο. Κάθε φεστιβάλ διαρκεί μία ή περισσότερες συνεχόμενες ημέρες και διαθέτει μοναδικά χαρακτηριστικά, όπως ημερομηνία, γεωγραφική θέση και πολιτισμικό πλαίσιο. Κάθε τοποθεσία περιγράφεται με ακρίβεια μέσω διεύθυνσης, συντεταγμένων, πόλης, χώρας και ηπείρου.

Η διοργάνωση περιλαμβάνει πολυάριθμες παραστάσεις (events), οι οποίες φιλοξενούνται σε προκαθορισμένες μουσικές σκηνές. Οι σκηνές εντάσσονται σε κτίρια που διαθέτουν τεχνικό εξοπλισμό (ηχεία, φώτα, μικρόφωνα, κονσόλες κ.ά.) και συνοδεύονται από πληροφορίες χωρητικότητας και περιγραφής. Κάθε σκηνή μπορεί να φιλοξενεί μόνο μία παράσταση ανά χρονική στιγμή.

Κάθε παράσταση αποτελείται από μία ή περισσότερες σειριακές εμφανίσεις καλλιτεχνών ή συγκροτημάτων. Οι εμφανίσεις περιγράφονται από τον τύπο συμμετοχής (π.χ. warm-up, headline, special guest), τη διάρκεια (μέχρι 3 ώρες), τη χρονική στιγμή και τον σχετιζόμενο καλλιτέχνη. Ανάμεσα σε συνεχόμενες εμφανίσεις εισάγεται διάλειμμα υποχρεωτικά διάρκειας από 5 έως 30 λεπτών.

Οι καλλιτέχνες του φεστιβάλ μπορεί να είναι σόλο ή μέλη μουσικών σχημάτων, με δυνατότητα ταυτόχρονης συμμετοχής σε πολλαπλά συγκροτήματα ή/και ατομικά. Για κάθε καλλιτέχνη καταγράφονται προσωπικά στοιχεία, καλλιτεχνικά ψευδώνυμα, μουσικά είδη και υποείδη, ημερομηνία γέννησης και διαθέσιμα ψηφιακά προφίλ (ιστοσελίδα, Instagram). Δεν επιτρέπεται η συμμετοχή ενός καλλιτέχνη ή συγκροτήματος σε δύο εμφανίσεις που διεξάγονται την ίδια χρονική στιγμή, ούτε η συμμετοχή τους στο φεστιβάλ για περισσότερα από τρία συνεχόμενα έτη.

Η λειτουργία του φεστιβάλ υποστηρίζεται από εξειδικευμένο προσωπικό, διαχωρισμένο σε τεχνικό, βοηθητικό και προσωπικό ασφαλείας, με διαφοροποίηση ανά ρόλο και επίπεδο εμπειρίας. Το πλήθος του προσωπικού σχετίζεται με τη χωρητικότητα της σκηνής: το προσωπικό ασφαλείας καλύπτει τουλάχιστον το 5% του κοινού και το βοηθητικό το 2%.

Οι επισκέπτες του φεστιβάλ μπορούν να αγοράσουν εισιτήρια ηλεκτρονικά, για συγκεκριμένες παραστάσεις και κατηγορίες (γενική είσοδος, VIP, backstage). Η πώληση υπόκειται σε αυστηρούς περιορισμούς: κάθε επισκέπτης μπορεί να έχει ένα εισιτήριο ανά ημέρα και παράσταση, η συνολική χωρητικότητα της σκηνής δεν μπορεί να ξεπεραστεί, ενώ τα VIP εισιτήρια περιορίζονται στο 10% της χωρητικότητας. Η πληρωμή επιτρέπεται μόνο μέσω τραπεζικών μέσων (χρεωστική/πιστωτική κάρτα ή τραπεζικός λογαριασμός).

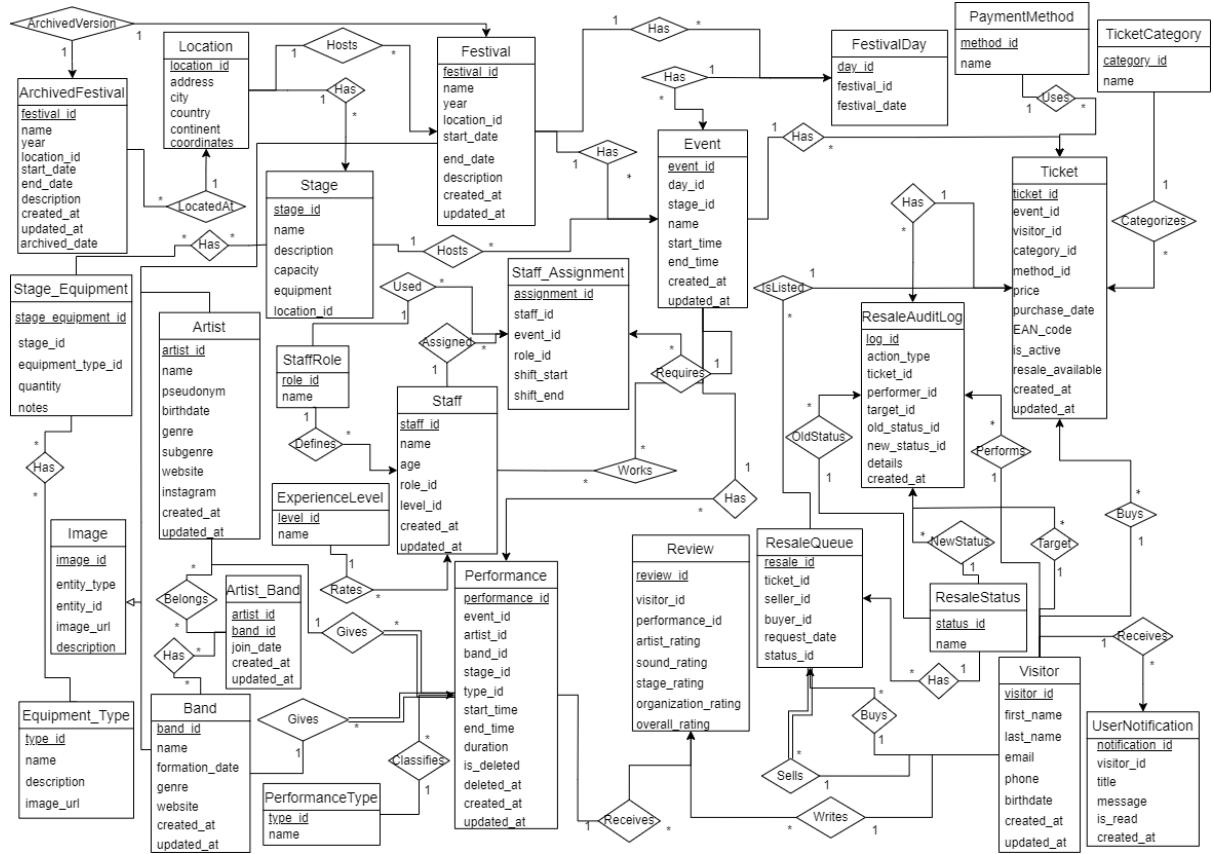
Σε περίπτωση εξάντλησης εισιτηρίων, ενεργοποιείται αυτόματα μηχανισμός μεταπώλησης, βασισμένος σε ουρές προτεραιότητας (FIFO), τόσο για τους πωλητές όσο και για τους ενδιαφερόμενους αγοραστές. Οι συναλλαγές πραγματοποιούνται αυτόματα, ενώ το εισιτήριο είναι δυνατό να μεταπωληθεί μόνο αν δεν έχει ενεργοποιηθεί.

Η αξιολόγηση των εμφανίσεων πραγματοποιείται αποκλειστικά από επισκέπτες που διαθέτουν ενεργοποιημένο εισιτήριο, με χρήση της κλίμακας Likert και βάσει πέντε κριτηρίων: ερμηνεία καλλιτεχνών, ήχος, μουσική σκηνή, οργάνωση και συνολική εντύπωση.

Ορισμένες βασικές οντότητες του συστήματος (όπως φεστιβάλ, καλλιτέχνες, εξοπλισμός) συνοδεύονται από οπτικό υλικό και σχετική περιγραφή, με στόχο τη μελλοντική ενσωμάτωσή τους σε διαδικτυακή εφαρμογή προβολής και διαχείρισης του φεστιβάλ.

## ΜΕΡΟΣ Α

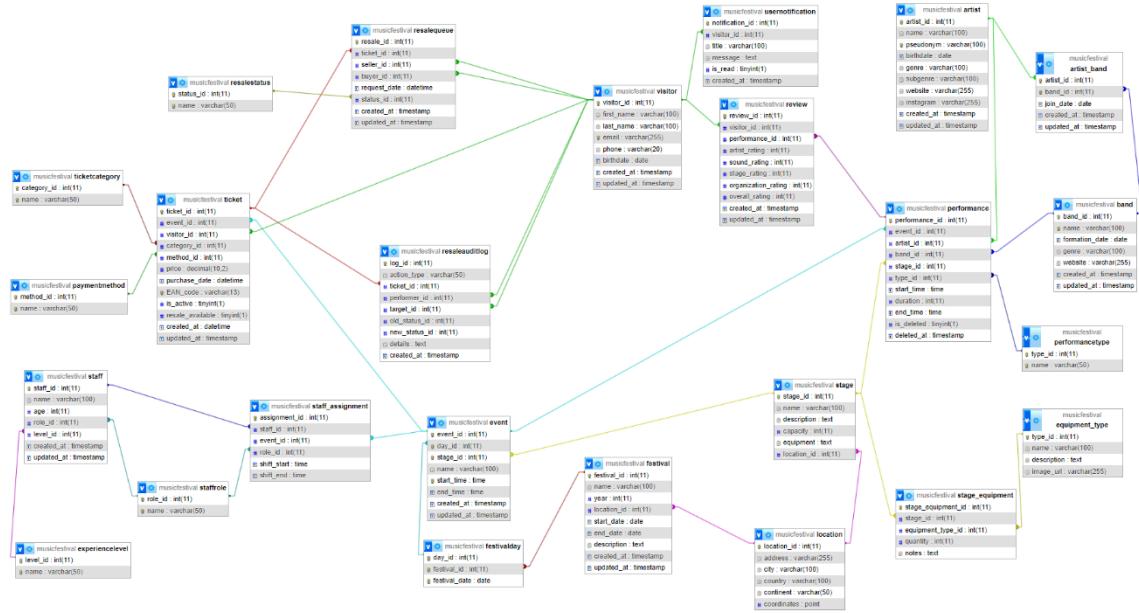
### ER Διάγραμμα



### Σχεσιακό Μοντέλο

Το παρακάτω σχεσιακό σχήμα αποτυπώνει τη λογική δομή της βάσης δεδομένων για το πληροφοριακό σύστημα του φεστιβάλ. Περιλαμβάνει βασικές οντότητες (όπως artist, event, ticket), σχέσεις μεταξύ τους, καθώς και βοηθητικούς πίνακες για την υποστήριξη επιπλέον λειτουργιών (π.χ. resalequeue, staff\_assignment, artist\_band). Το σχήμα ακολουθεί κανονικοποιημένη δομή και σχεδιάστηκε με γνώμονα την ακεραιότητα, τη συνέπεια και την επεκτασιμότητα του συστήματος.

## Σχήμα:



## Υποσημειώσεις Τύπων Σχέσεων:

- 1:1** – Ένα στοιχείο του πίνακα A σχετίζεται με το πολύ ένα στοιχείο του πίνακα B και αντίστροφα.
- 1:N** – Ένα στοιχείο του πίνακα A μπορεί να σχετίζεται με πολλά στοιχεία του πίνακα B.
- N:1** – Πολλά στοιχεία του πίνακα A σχετίζονται με ένα στοιχείο του πίνακα B.
- N:M** – Πολλά στοιχεία του πίνακα A σχετίζονται με πολλά του πίνακα B μέσω ενδιάμεσου πίνακα.

## Υλοποίηση της Βάσης Δεδομένων σε SQL

Η βάση δεδομένων του Pulse University Music Festival υλοποιήθηκε σε MySQL/MariaDB, αξιοποιώντας SQL δυνατότητες για την εξασφάλιση ακεραιότητας, συνέπειας και αποδοτικότητας. Η σχεδίαση βασίστηκε στο σχεσιακό μοντέλο, ακολουθώντας αρχές κανονικοποίησης και ορθής διαχείρισης επιχειρησιακής λογικής μέσω περιορισμών (constraints), triggers, stored procedures και views, ευρετηρίων και συναρτήσεων.

### 1. Δημιουργία της Βάσης

Η βάση δεδομένων δημιουργήθηκε με εντολές:

```
CREATE DATABASE MusicFestival;
```

```
USE MusicFestival;
```

### 2. Πίνακες Αναφοράς (Lookup Tables)

Οι πίνακες αναφοράς περιλαμβάνουν σταθερές τιμές που χρησιμοποιούνται από άλλους πίνακες για τη διασφάλιση ακεραιότητας δεδομένων, αναγνωσιμότητας και επεκτασιμότητας. Όλοι οι πίνακες αυτού του τύπου ακολουθούν κοινή δομή:

- Πρωτεύον Κλειδί:** ακέραιος κωδικός (AUTO\_INCREMENT)
- Όνομα:** περιγραφικό πεδίο (VARCHAR, UNIQUE, NOT NULL)
- Μεταδεδομένα (προαιρετικά):** δυνατότητα προσθήκης πεδίων όπως description ή created\_at στο μέλλον

Πίνακας	Πεδία	Περιγραφή	Χρήση
PaymentMethod	method_id (PK), name	Τρόποι πληρωμής: "credit_card", "debit_card", "bank_transfer".	Χρησιμοποιούνται για τον προσδιορισμό πληρωμής (Ticket, Payment).
ExperienceLevel	level_id (PK), name	Επίπεδα εμπειρίας προσωπικού: "intern", "beginner", "intermediate"	Προσδιορίζει τον βαθμό εμπειρίας κάθε μέλους (Staff).
PerformanceType	type_id (PK), name	Είδος εμφάνισης καλλιτέχνη: "warm-up", "headline", "guest".	Ορίζει τον ρόλο του καλλιτέχνη (Performance).

<b>StaffRole</b>	role_id (PK), name	Ρόλος προσωπικού: "technical", "security", "support".	Κατηγοριοποίηση καθηκόντων (Staff, Staff_Assignment).
<b>ResaleStatus</b>	status_id (PK), name	Κατάσταση αιτήματος μεταπώλησης: "Available", "Sold", "Pending", "Cancelled".	Τρέχουσα κατάσταση (ResaleQueue).
<b>TicketCategory</b>	category_id (PK), name	Τύποι εισιτηρίων: "general", "VIP", "backstage".	Καθορίζει την κατηγορία πρόσβασης (Ticket).

### 3. Πίνακες Οντοτήτων

Αποτελούν τον πυρήνα της βάσης δεδομένων, αποτυπώνοντας τις βασικές οντότητες που συμμετέχουν στη λειτουργία του φεστιβάλ.

#### Location

Αποθηκεύει γεωγραφικά δεδομένα τοποθεσιών φεστιβάλ και εγκαταστάσεων.

**Πεδία:** location\_id (PK), address, city, country, continent, coordinates (POINT, με SPATIAL INDEX)

Συνδέεται με τους πίνακες Festival, Stage και Staff, υποστηρίζοντας αποδοτικά γεωγωρικά queries (π.χ. αναζήτηση κοντινών σκηνών).

#### Festival

Αποτυπώνει κάθε ετήσιο φεστιβάλ.

**Πεδία:** festival\_id (PK), name, year, start\_date, end\_date, description, location\_id (FK)

**Περιορισμοί:** start\_date ≤ end\_date, YEAR(start\_date) = year

Συσχετίζεται με FestivalDay, Event και Location.

#### Stage

Αντιπροσωπεύει τις μουσικές σκηνές.

**Πεδία:** stage\_id (PK), name, capacity, description, equipment, location\_id (FK)

**Περιορισμοί:** capacity > 0

Συσχετίζεται με Event, Performance, Stage\_Equipment.

## **FestivalDay**

Ορίζει τις ημέρες ενός φεστιβάλ.

**Πεδία:** day\_id (PK), festival\_id (FK), festival\_date

**Περιορισμοί:**

- Trigger CheckFestivalYear: festival\_date στο year του φεστιβάλ
- ON DELETE CASCADE, εξασφαλίζει ότι σε διαγραφή ενός φεστιβάλ, διαγράφονται αυτόματα και όλες οι αντίστοιχες εγγραφές στο FestivalDay.
- UNIQUE (festival\_id, festival\_date), διασφαλίζει ότι η ίδια ημερομηνία δεν μπορεί να καταχωρηθεί δύο φορές για το ίδιο φεστιβάλ.

## **Event**

Καταγράφει παραστάσεις σε συγκεκριμένες σκηνές και ημέρες.

**Πεδία:** event\_id (PK), day\_id (FK), stage\_id (FK), name, start\_time, end\_time, created\_at, updated\_at

**Περιορισμοί:**

- CHECK (start\_time < end\_time)
- UNIQUE (day\_id, stage\_id, start\_time), προτρέπει επικαλύψεις δηλαδή σε δύο events να ξεκινούν την ίδια στιγμή στην ίδια σκηνή την ίδια μέρα.
- Trigger check\_overlapping\_events
- ON DELETE CASCADE στα FK, ώστε αν διαγραφεί μια μέρα ή μια σκηνή, να διαγράφονται και τα events που σχετίζονται με αυτή.

Συγχετίζεται με FestivalDay και Stage.

## **Artist**

Αποθηκεύει στοιχεία μεμονωμένων καλλιτεχνών.

**Πεδία:** artist\_id (PK), name, pseudonym (UNIQUE-μοναδικό ανά καλλιτέχνη), birthdate, genre, subgenre, instagram, website, created\_at, updated\_at

**Περιορισμός:** uq\_artist\_pseudonym για μοναδικότητα pseudonym.

## **Band**

Καταγράφει μουσικά συγκροτήματα.

**Πεδία:** band\_id (PK), name, formation\_date, genre, website, created\_at, updated\_at

Συγχετίζεται με Artist μέσω του Artist\_Band.

## Artist Band

Υλοποιεί many-to-many σχέση καλλιτεχνών και συγκροτημάτων.

**Πεδία:** artist\_id (FK), band\_id (FK), join\_date, created\_at, updated\_at

**Περιορισμοί:**

- Πρωτεύον σύνθετο κλειδί: (artist\_id, band\_id), διασφαλίζεται ότι ένας καλλιτέχνης δεν μπορεί να καταχωρηθεί δύο φορές στο ίδιο συγκρότημα.
- ON DELETE CASCADE στα FK, αν διαγραφεί ένας καλλιτέχνης ή μπάντα, να διαγράφονται αυτόματα οι εγγραφές συμμετοχής του σε συγκροτήματα .

## Visitor

Καταγράφει πληροφορίες των επισκεπτών στο φεστιβάλ.

**Πεδία:** visitor\_id (PK), first\_name, last\_name, email, phone, birthdate, created\_at, updated\_at

**Περιορισμοί & Trigger:**

- check\_email\_format: έλεγχος εγκυρότητας email με REGEXP
- check\_visitor\_age\_insert & check\_visitor\_age\_update: διασφαλίζουν ότι η ηλικία είναι  $\geq 16$  ετών κατά την αγορά εισιτηρίου
- UNIQUE (email), διασφαλίζεται ότι το ίδιο email δεν μπορεί να χρησιμοποιηθεί σε παραπάνω από έναν επισκέπτη.

## Staff

Αντιπροσωπεύει τα στελέχη και τεχνικό προσωπικό.

**Πεδία:** staff\_id (PK), name, role\_id (FK), age, experience\_level\_id (FK), location\_id (FK), created\_at, updated\_at.

**Περιορισμοί:** CHECK (age  $\geq 18$ ) για διασφάλιση ελάχιστης ηλικίας

Σχετίζεται με StaffRole, ExperienceLevel, Location.

## Staff Assignment

Καταγράφει αναθέσεις προσωπικού σε events και σκηνές.

**Πεδία:** assignment\_id (PK), staff\_id (FK), event\_id (FK), role\_id (FK), shift\_start, shift\_end

**Triggers:**

- CheckSecurityStaff: Ελέγχει ότι καλύπτεται τουλάχιστον το 5% της χωρητικότητας από προσωπικό ασφαλείας.
- CheckSupportStaff: Ελέγχει ότι καλύπτεται τουλάχιστον το 2% από υποστηρικτικό προσωπικό.
- ON DELETE CASCADE σε staff\_id και event\_id ώστε διαγραφές σε Staff ή Event να διαγράφουν και τις αντίστοιχες αναθέσεις.

Σχετίζεται με StaffRole, Staff, StaffRole, και Event.

### **Performance**

Καταγράφει κάθε εμφάνιση καλλιτέχνη/συγκροτήματος σε ένα event.

**Πεδία:** performance\_id (PK), event\_id (FK), artist\_id (nullable FK), band\_id (nullable FK), stage\_id (FK), type\_id (FK), start\_time, end\_time((PERSISTENT) που προκύπτει από το start\_time + duration.), duration, is\_deleted (default FALSE), deleted\_at

- Τα εξής ξένα κλειδία έχουν τους αντίστοιχους περιορισμούς FOREIGN KEY artist\_id με ON DELETE SET NULL για να μην διαγράφεται η εμφάνιση αν διαγραφεί ο καλλιτέχνης και για το stage\_id με ON DELETE CASCADE.

### **Περιορισμοί:**

- CHECK (artist\_id IS NOT NULL OR band\_id IS NOT NULL), δηλαδή τουλάχιστον ένα από τα δύο πρέπει να είναι NOT NULL
- CHECK (duration BETWEEN 1 AND 180)

Σχετίζεται με Event, Artist, Band, Stage, PerformanceType

### **Ticket**

Καταγραφή εισιτηρίων ανά επισκέπτη και εμφάνιση.

- Πεδία: ticket\_id (PK), visitor\_id (FK), event\_id (FK), category\_id (FK), method\_id (FK), price, EAN\_code, status, purchase\_date, is\_active, resale\_available, created\_at (DEFAULT CURRENT\_TIMESTAMP), updated\_at, (ON UPDATE CURRENT\_TIMESTAMP- αυτόματη ενημέρωση όταν τροποποιηθεί)
- Χρησιμοποιούνται τα εξής Triggers:
  - CheckVIPTicketLimit: VIP ≤ 10% χωρητικότητας

- CheckOneTicketPerVisitorPerDayAndPerformance: Ένας επισκέπτης, ένα εισιτήριο ανά performance ανά ημέρα

Σχετίζεται με Event, Visitor, TicketCategory, PaymentMethod.

### **ResaleQueue**

Χρησιμοποιείται στο Σύστημα Μεταπώλησης Εισιτηρίων με Ουρά FIFO ώστε να κάνει καταγραφή αιτημάτων

**Πεδία:** resale\_id (PK), ticket\_id (FK), seller\_id (FK), buyer\_id (FK), status\_id (FK), request\_date, created\_at (DEFAULT CURRENT\_TIMESTAMP), updated\_at, (ON UPDATE CURRENT\_TIMESTAMP- αυτόματη ενημέρωση όταν τροποποιηθεί).

- Το πεδίο request\_date δημιουργήθηκε για την εφαρμογή της λογικής FIFO
- Ο πίνακας αναφοράς ResaleStatus χρησιμοποιείται για διαχείριση καταστάσεων (Available, Pending, Sold, Cancelled)

**Triggers & Procedures:** αυτοματοποιούν τη λειτουργικότητα της ουράς

- CheckResaleAvailability: Επιτρέπει μεταπώληση μόνο αν είναι ενεργή
- ProcessResaleQueue: Εκτελεί λογική μεταπώλησης με βάση FIFO
- ON DELETE CASCADE: Αν διαγραφεί το εισιτήριο ή οι επισκέπτες, διαγράφονται και οι μεταπωλήσεις που τους αφορούν.

Σχετίζεται με Ticket, Visitor, ResaleStatus.

### **Review**

Καταγραφή αξιολογήσεων επισκεπτών για εμφανίσεις.

**Πεδία:** review\_id (PK), visitor\_id (FK), performance\_id (FK), artist\_rating, sound\_rating, stage\_rating, created\_at, updated\_at.

**Περιορισμός:** CHECK, για διασφάλιση τιμών μεταξύ 1-5

**Stored Procedure:** InsertReview(): Επιτρέπει εισαγωγή μόνο αν υπάρχει ενεργό εισιτήριο για την εμφάνιση

Τα ξένα κλειδιά χρησιμοποιούν ON DELETE CASCADE όπου αν διαγραφεί ο επισκέπτης, διαγράφονται και οι κριτικές του και αντίστοιχα αν διαγραφεί η εμφάνιση, διαγράφονται και οι κριτικές της.

Συνδέεται με τους πίνακες Visitor και Performance.

### Image

Αποθήκευση URL εικόνων που συνδέονται με άλλες οντότητες.

**Πεδία:** image\_id (PK), entity\_type, entity\_id, image\_url, description

**Περιορισμοί:** CHECK, entity\_type IN ('artist', 'band', 'stage', 'festival') όπου ελέγχει ότι το entity\_type παίρνει αποδεκτές τιμές

## 4. Triggers

Για τη διασφάλιση της ακεραιότητας και την αυτοματοποίηση κρίσιμων επιχειρησιακών κανόνων, υλοποιήθηκαν triggers που ενεργοποιούνται σε εισαγωγές, ενημερώσεις και διαγραφές. Ελέγχουν χρονικά όρια, περιορισμούς συμμετοχής, πληρότητα και όρους αγοράς, διασφαλίζοντας συνέπεια και αυτονομία της βάσης χωρίς εξωτερική λογική.

Όνομα Trigger	Περιγραφή
<b>CheckFestivalYear</b>	Διασφαλίζει ότι η festival_date είναι εντός του έτους του αντίστοιχου Festival.
<b>check_overlapping_events</b>	Ακυρώνει νέα Event αν υπάρχει χρονική επικάλυψη στην ίδια σκηνή και ημέρα.
<b>CheckConcurrentPerformances</b>	Αποτρέπει παράλληλες εμφανίσεις του ίδιου καλλιτέχνη/συγκροτήματος.
<b>CheckPerformanceGap</b>	Επιβάλλει διάστημα 5–30 λεπτών μεταξύ παραστάσεων στην ίδια σκηνή.
<b>CheckConsecutiveYears</b>	Αποτρέπει 4η συνεχόμενη ετήσια συμμετοχή του ίδιου καλλιτέχνη/συγκροτήματος.
<b>CheckVIPTicketLimit</b>	Περιορίζει τα VIP εισιτήρια στο 10% της χωρητικότητας κάθε σκηνής.
<b>CheckVisitorAgeForTicketPurchase</b>	Ακυρώνει αγορά εισιτηρίου από επισκέπτες κάτω των 16 ετών.

<b>check_visitor_age_insert</b>	Αποτρέπει την εισαγωγή επισκέπτη κάτω των 16 ετών.
<b>check_visitor_age_update</b>	Αποτρέπει ενημέρωση ηλικίας σε τιμή κάτω των 16 ετών.
<b>CheckOneTicketPerVisitor PerDayAndPerformance</b>	Επιτρέπει μόνο ένα εισιτήριο ανά επισκέπτη για κάθε ημερομηνία και event.
<b>DeactivateUsedTicket</b>	Ενημερώνει την κατάσταση εισιτηρίου ώστε να μην επιτρέπεται μεταπώληση.
<b>CheckSecurityStaff</b> <b>CheckSupportStaff</b>	Εξασφαλίζει ελάχιστο προσωπικό ασφαλείας (5%) και βοηθητικό (2%) ανά event/stage.
<b>ProcessResaleQueue</b>	Αντιστοιχίζει εισιτήριο προς μεταπώληση στον πρώτο χρήστη σε αναμονή (FIFO).
<b>CheckResaleAvailability</b>	Αποτρέπει μεταπωλήσεις μη διαθέσιμων εισιτηρίων (resale_available = FALSE).
<b>CheckEANFormat</b>	Ελέγχει αν ο EAN_code έχει ακριβώς 13 ψηφία (EAN-13).
<b>reset_payment_method_id</b>	Καταγράφει διαγραφή όλων των τρόπων πληρωμής.
<b>reset_experience_level_id</b>	Καταγράφει διαγραφή όλων των επιπέδων εμπειρίας.
<b>reset_performance_type_id</b>	Καταγράφει διαγραφή όλων των τύπων παραστάσεων.
<b>reset_staff_role_id</b>	Καταγράφει διαγραφή όλων των ρόλων προσωπικού.
<b>reset_resale_status_id</b>	Καταγράφει διαγραφή όλων των καταστάσεων μεταπώλησης.
<b>reset_ticket_category_id</b>	Καταγράφει διαγραφή όλων των κατηγοριών εισιτηρίων.
<b>reset_location_id</b>	Καταγράφει διαγραφή όλων των τοποθεσιών.
<b>reset_festival_id</b>	Καταγράφει πλήρη απώλεια φεστιβάλ.
<b>reset_stage_id</b>	Καταγράφει διαγραφή όλων των σκηνών.
<b>reset_festival_day_id</b>	Καταγράφει όταν δεν υπάρχουν ημέρες φεστιβάλ.
<b>reset_event_id</b>	Καταγράφει πλήρη διαγραφή των event.

<b>reset_artist_id</b>	Καταγράφει διαγραφή όλων των καλλιτεχνών.
<b>reset_band_id</b>	Καταγράφει διαγραφή όλων των συγκροτημάτων.
<b>reset_visitor_id</b>	Καταγράφει διαγραφή όλων των επισκεπτών.
<b>reset_staff_assignment_id</b>	Καταγράφει διαγραφή από Staff_Assignment και προτείνει reset AUTO_INCREMENT.
<b>reset_staff_id</b>	Καταγράφει διαγραφή όλων των αναθέσεων προσωπικού.
<b>reset_performance_id</b>	Ενεργοποιείται όταν δεν υπάρχουν ενεργές (μη διαγραμμένες) παραστάσεις.
<b>reset_review_id</b>	Καταγράφει διαγραφή όλων των κριτικών.
<b>reset_ticket_id</b>	Καταγράφει διαγραφή όλων των εισιτηρίων.
<b>reset_resale_queue_id</b>	Καταγράφει διαγραφή όλων των εγγραφών στη ResaleQueue.
<b>reset_image_id</b>	Καταγράφει διαγραφή όλων των εικόνων.
<b>reset_type_id</b>	Καταγράφει όταν αδειάσει ο πίνακας Equipment_Type.
<b>stage_equipment_id</b>	Όταν ο Stage_Equipment αδειάσει, προτείνει reset AUTO_INCREMENT.

### Περιορισμός Συνεχόμενων Συμμετοχών για 3 έτη - CheckConsecutiveYears

**Πρόβλημα:** Ο περιορισμός "οι καλλιτέχνες δεν μπορούν να συμμετέχουν για περισσότερα από 3 συνεχόμενα έτη" ήταν πολύπλοκος για υλοποίηση, καθώς απαιτούσε λογική που εκτείνεται σε δεδομένα πολλαπλών ετών.

**Λύση:** Αναπτύξαμε έναν σύνθετο trigger ώστε:

1. Για κάθε νέα εμφάνιση, υπολογίζει το έτος του φεστιβάλ.
2. Αναζητά παρουσίες του ίδιου καλλιτέχνη/συγκροτήματος στα δύο προηγούμενα έτη.
3. Αν ο καλλιτέχνης έχει εμφανιστεί και στα δύο προηγούμενα συνεχόμενα έτη, απορρίπτει την εισαγωγή.
4. Χειρίζεται διαφορετικά τους καλλιτέχνες και τα συγκροτήματα με παρόμοια λογική.

Αυτή η προσέγγιση:

- Διατηρεί τη βασική λογική στη βάση δεδομένων, εξασφαλίζοντας ακεραιότητα ανεξάρτητα από την εφαρμογή πελάτη
- Επιτρέπει τη συμμετοχή μετά από έτος αποχής, καθώς ελέγχει μόνο για τρία συνεχόμενα έτη
- Υποστηρίζει αποτελεσματικά τα ερωτήματα #11 και #13

### Διαλείμματα Μεταξύ Εμφανίσεων (5-30 λεπτά) - CheckPerformanceGap

**Πρόβλημα:** Η απαίτηση για διαλείμματα μεταξύ 5-30 λεπτών μεταξύ διαδοχικών εμφανίσεων στην ίδια σκηνή αποτελούσε πρόκληση, καθώς απαιτεί τον έλεγχο χρονικών διαστημάτων μεταξύ εγγραφών.

**Λύση:** Υλοποιήσαμε τον trigger ώστε:

1. Για κάθε νέα εμφάνιση, εντοπίζει την προηγούμενη και την επόμενη εμφάνιση στην ίδια σκηνή και ημέρα
2. Υπολογίζει το χρονικό διάστημα πριν και μετά τη νέα εμφάνιση
3. Ελέγχει αν τα διαστήματα είναι εντός των απαιτούμενων ορίων (5-30 λεπτά)
4. Απορρίπτει την εισαγωγή αν παραβιάζονται τα όρια

Αυτή η προσέγγιση:

- Εξασφαλίζει την τήρηση του περιορισμού διαλειμμάτων
- Επιτρέπει τον αποτελεσματικό προγραμματισμό παραστάσεων
- Υποστηρίζει τα ερωτήματα #3 και #14 που αναλύουν μοτίβα εμφανίσεων

## 5. Stored Procedures

Οι Stored Procedures είναι προκαθορισμένες υπορουτίνες στον SQL server που επιτελούν επαναχρησιμοποιήσιμες λειτουργίες επιχειρησιακής λογικής, διασφαλίζοντας ακεραιότητα, απόδοση και ασφάλεια. Χρησιμοποιούνται αυτόνομα ή εντός triggers/events, περιορίζοντας την πολυπλοκότητα της εφαρμογής.

### InsertReview

Εισάγει κριτική σε παράσταση εφόσον ο επισκέπτης έχει χρησιμοποιημένο εισιτήριο. Διαφορετικά, επιστρέφεται μήνυμα σφάλματος.

### process\_ticket\_resale

Εκτελεί μεταπώληση εισιτηρίου εάν είναι διαθέσιμο. Αναθέτει νέο αγοραστή, ενημερώνει τη διαθεσιμότητα, καταγράφει τη συναλλαγή και κάνει COMMIT. Διαφορετικά, τερματίζεται σιωπηλά.

## **archive\_old\_festivals**

Αρχειοθετεί φεστιβάλ παλαιότερα των 5 ετών στον πίνακα ArchivedFestival, χωρίς διαγραφή των αρχικών εγγραφών.

## **update\_festival\_statistics\_cache**

Υπολογίζει και αποθηκεύει στατιστικά φεστιβάλ σε JSON μορφή στον πίνακα QueryCache. Αν υπάρχει ήδη το cache\_key, γίνεται ενημέρωση. Διαφορετικά, εισάγεται νέα εγγραφή με ημερομηνία λήξης +1 μέρα.

## **list\_ticket\_for\_resale**

Ελέγχει αν εισιτήριο είναι ενεργό, αχρησιμοποίητο και εντός ορίου χρόνου. Αν ισχύει, δημιουργείται εγγραφή μεταπώλησης με status 'pending'. Άλλιώς απορρίπτεται με μήνυμα.

## **soft\_delete\_performance**

Εκτελεί λογική διαγραφή (soft delete) εμφάνισης μέσω ενημέρωσης is\_deleted = TRUE και καταγραφής deleted\_at.

## **search\_artists**

Αναζητά καλλιτέχνες μέσω full-text search βάσει ονόματος, είδους ή υποείδους και επιστρέφει αποτελέσματα κατά σχετικότητα.

## **process\_reset\_log**

Χρησιμοποιεί cursor για επαναφορά AUTO\_INCREMENT σε πίνακες με καταγραφή σχετικής ενέργειας στο ResetLog. Διαγράφει την εγγραφή μετά την επεξεργασία.

## **request\_to\_buy\_ticket**

Εκτελεί αίτημα αγοράς μεταπωλούμενου εισιτηρίου. Πραγματοποιεί ελέγχους εγκυρότητας, διαθεσιμότητας και ειδοποιεί τον πωλητή. Σε επιτυχία γίνεται COMMIT, αλλιώς ROLLBACK.

## **view\_pending\_requests**

Παρέχει στον πωλητή λίστα με αιτήματα αγοράς των εισιτηρίων του που είναι σε 'Pending' κατάσταση, με λεπτομερή πληροφορία ανά αίτημα.

## **approve\_purchase\_request**

Ολοκληρώνει μεταπώληση εισιτηρίου. Εγκρίνει το αίτημα, μεταφέρει το εισιτήριο, ειδοποιεί τον αγοραστή, ακυρώνει άλλα αιτήματα για το ίδιο εισιτήριο και καταγράφει τις ενέργειες.

### **reject\_purchase\_request**

Ακυρώνει συγκεκριμένο αίτημα αγοράς, ειδοποιεί τον αγοραστή και αν δεν υπάρχουν άλλα αιτήματα, προσθέτει νέα διαθέσιμη καταχώριση στο ResaleQueue.

### **cancel\_expired\_requests**

Ακυρώνει αιτήματα που είναι 'Pending' για >24 ώρες, επαναφέρει εισιτήρια ως διαθέσιμα αν δεν υπάρχουν άλλα αιτήματα και ειδοποιεί τους χρήστες. Εκτελείται με ασφάλεια μέσω transaction.

### **mark\_notification\_read**

Ορίζει ειδοποίηση ως αναγνωσμένη για συγκεκριμένο χρήστη, ενημερώνοντας το πεδίο is\_read.

### **get\_visitor\_notifications(IN p\_visitor\_id INT, IN p\_include\_read BOOLEAN)**

Επιστρέφει τις ειδοποίησεις ενός επισκέπτη από τον πίνακα UserNotification, με επιλογή συμπερίληψης ή όχι των αναγνωσμένων. Επιστρέφει notification\_id, title, message, is\_read, created\_at, ταξινομημένα κατά φθίνουσα ημερομηνία.

### **get\_ticket\_audit\_log(IN p\_ticket\_id INT)**

Παρέχει το ιστορικό ενεργειών για ένα εισιτήριο από τον πίνακα ResaleAuditLog. Περιλαμβάνει log\_id, action\_type, εκτελεστή και αποδέκτη ενέργειας, αλλαγές κατάστασης, λεπτομέρειες και created\_at, ταξινομημένα κατά φθίνουσα ημερομηνία.

## **6. Views (Όψεις)**

Οι όψεις είναι εικονικοί πίνακες που βασίζονται σε σύνθετα ερωτήματα SQL και χρησιμοποιούνται για να απλοποιήσουν την πρόσβαση σε δεδομένα, να υποστηρίξουν αναφορές και να ενισχύσουν την ασφάλεια και επαναχρησιμοποίηση λογικής.

Όνομα Όψης	Περιγραφή	Χρήση
<b>artist_with_age</b>	Επιστρέφει τους καλλιτέχνες με δυναμικά υπολογισμένη ηλικία βάσει της birthdate.	Χρήσιμο για φιλτράρισμα (π.χ. ηλικίες < 30), αναλύσεις και UI προβολές.
<b>active_performances</b>	Περιλαμβάνει τις εμφανίσεις από Performance όπου is_deleted = FALSE.	Προβολή μόνο ενεργών εμφανίσεων σε πρόγραμμα ή frontend.
<b>active_events</b>	Περιλαμβάνει παραστάσεις που συνδέονται με φεστιβάλ που δεν	Εμφάνιση μελλοντικών ή τρεχουσών εκδηλώσεων για πωλήσεις και διαχείριση.

	έχουν ακόμα ολοκληρωθεί (festival_date >= CURDATE()).	
--	--	--

## 7. Ευρετήρια (Indexes)

Η σχεδίαση των ευρετηρίων έγινε με στόχο τη βελτιστοποίηση των ερωτημάτων που προσδιορίζονται στην εκφώνηση. Συγκεκριμένα:

### Βασικά Ευρετήρια για Πρωτεύοντα Κλειδιά

Αυτόματα δημιουργούνται για όλα τα πρωτεύοντα κλειδιά (π.χ. PRIMARY στον πίνακα Artist)

**Ευρετήρια:**

Ευρετήριο	Πίνακας	Πεδία	Query	Αιτιολόγηση
idx_artist_genre	Artist	genre	#2, #10	Επιταχύνει την αναζήτηση καλλιτεχνών ανά μουσικού είδους
idx_artist_birthdate	Artist	birthdate	#5	Βελτιστοποιεί την εύρεση νέων καλλιτεχνών (ηλικία < 30)
idx_performance_artist	Performance	artist_id	#3, #4, #11, #13	Κρίσιμο για ερωτήματα που συνδέουν καλλιτέχνες με παραστάσεις
idx_performance_band	Performance	band_id	#13	Επιταχύνει τη σύνδεση συγκροτημάτων με παραστάσεις
idx_review_performance	Review	performance_id	#4	Βασικό για την εύρεση αξιολογήσεων ανά παράσταση
idx_review_visitor	Review	visitor_id	#6, #15	Κρίσιμο για ερωτήματα που φιλτράρουν αξιολογήσεις ανά επισκέπτη
idx_ticket_visitor	Ticket	visitor_id	#6, #9	Επιταχύνει την εύρεση εισιτηρίων ανά επισκέπτη
idx_ticket_event	Ticket	event_id	#1, #12	Βελτιστοποιεί ερωτήματα σχετικά με έσοδα και συμμετοχές ανά εκδήλωση
idx_location_coordinates	Location	coordinates	#13	SPATIAL ευρετήριο για γεωγραφικές αναζητήσεις,

				χρήσιμο για ερωτήματα ανά ήπειρο
idx_staff_role idx_staff_assignment_role	Staff	role_id	#7, #8, #12	Βασικό για φίλτραρισμα προσωπικού με βάση το ρόλο
ft_artist_search	Artist	name, pseudonym, genre, subgenre	#10	FULLTEXT ευρετήριο για αναλυτική αναζήτηση μουσικών ειδών και συνδυασμών
idx_resale_ticket	ResaleQueue	ticket_id	-	Βελτιστοποιεί το σύστημα μεταπώλησης εισιτηρίων FIFO
idx_resale_status	ResaleQueue	status_id	-	Επιταχύνει φίλτραρισμα εισιτηρίων ανά κατάσταση μεταπώλησης
idx_performance_start	Performance	start_time	-	Κρίσιμο για έλεγχο διαστημάτων και επικαλύψεων παραστάσεων

### Οφέλη:

- Ταχύτερο Φίλτραρισμα: Μειώνει την αδράνεια ερωτημάτων με WHERE (π.χ., φίλτρα ανά genre, ημερομηνία).
- Αποδοτικοί Σύνδεσμοι (JOINS): Βελτιστοποιεί συνδέσεις πινάκων με βάση ξένα κλειδιά (π.χ., artist\_id ↔ Performance).
- Προηγμένη Αναζήτηση: Χωρικά και πλήρους κειμένου ευρετήρια υποστηρίζουν πολύπλοκα σενάρια.
- Κλιμακωσιμότητα: Διασφαλίζει απόδοση ακόμα και με μεγάλο όγκο δεδομένων.

### 8. Υποσυστήματα και Υποστηρικτικοί Πίνακες

Για την κάλυψη ειδικών απαιτήσεων, υλοποιήθηκαν βοηθητικά υποσυστήματα που υποστηρίζουν caching, auditing, automation και ειδοποιήσεις. Αυτά ενισχύουν τη λειτουργικότητα, διαχειρισιμότητα και απόδοση του συστήματος.

Όνομα Πίνακα / Υποσυντήματος	Πεδία / Δομή	Ρόλος / Περιγραφή	Χρήση
<b>QueryCache</b>	cache_id (PK), entity_type, entity_id, payload (JSON), last_updated	Προσωρινή αποθήκευση προϋπολογισμένων δεδομένων (payload ως JSON).	Επιτάχυνση dashboards και reports.
<b>ResaleAuditLog</b>	audit_id (PK), resale_id (FK), action, actor_id, timestamp,details(JSON)	Ιστορικό αλλαγών στη μεταπώληση εισιτηρίων. Περιλαμβάνει actor, action και λεπτομέρειες.	Debugging, auditing, διαφύνεια ενεργειών.
<b>UserNotification</b>	notification_id (PK), recipient_id,recipient_type,message,delivery_type, is_read, created_at	Υποδομή αποστολής και παρακολούθησης ειδοποίησεων (in-app, email, SMS).	Alerts για χρήστες, αλλαγές, updates.
<b>ArchivedFestival</b>	festival_id (PK), name, year,start_date, end_date, description, location_id (FK)	Ιστορικό φεστιβάλ που δεν επηρεάζει ενεργές λειτουργίες.	Auditing και ιστορική αναφορά.
<b>Equipment_Type</b>	type_id (PK), name, description, image_url	Κατάλογος διαθέσιμου εξοπλισμού με όνομα, περιγραφή & εικόνα.	Χρήση από άλλους πίνακες (Stage_Equipment)
<b>Stage_Equipment</b>	stage_equipment_id, stage_id,equipment_type_id, quantity, notes	Συσχέτιση σκηνής με τύπους εξοπλισμού και ποσότητες.	Διαχείριση υλικοτεχνικής υποδομής φεστιβάλ.

## 9. Σχολιασμός Εισαγωγής Δεδομένων

Το αρχείο load.sql περιλαμβάνει την αρχικοποίηση και εισαγωγή βασικών δεδομένων για την ορθή λειτουργία της βάσης, ακολουθώντας το σχεσιακό σχήμα που ορίστηκε.

### 1) Εισαγωγή στατικών οντοτήτων

Πίνακες όπως PaymentMethod, TicketCategory, ResaleStatus, PerformanceType, StaffRole, ExperienceLevel, Equipment\_Type γεμίζονται με σταθερά δεδομένα με απλές INSERT INTO εντολές. Αυτά τα δεδομένα:

Παρέχουν αρχικές επιλογές στα dropdowns της εφαρμογής.

Υποστηρίζουν ελέγχους integrity μέσω foreign keys (π.χ. status\_id στην resalequeue).

Επιτρέπουν την τυποποίηση και επεκτασιμότητα.

### **Παράδειγμα:**

```
INSERT INTO PaymentMethod (method_id, name) VALUES (1, 'Credit Card'), (2, 'PayPal'),  
(3, 'Cash');
```

### **2) Εισαγωγή κύριων οντοτήτων (main entities)**

Γίνεται προσεκτική εισαγωγή σε πίνακες Artist, Band, Visitor, Staff, Stage, Festival, Event, Performance. Οι εντολές ακολουθούν τη σωστή σειρά εισαγωγής, ώστε να μην παραβιαστούν οι περιορισμοί των ξένων κλειδιών, π.χ. πρώτα εισάγονται Location, μετά Festival, και μετά Stage με location\_id.

### **3) Εισαγωγή πολυπλοκότερων εγγραφών**

Η εισαγωγή για τους εξής πίνακες:

Staff\_Assignment: περιλαμβάνει συσχέτιση staff–event–role με shift.

Artist\_Band: χειρίζεται N:M σχέση μεταξύ Artist και Band.

Performance: γίνεται χειρισμός nullable artist\_id / band\_id.

Γίνεται εφαρμογή περιορισμών CHECK και foreign keys.

### **4) Χρήση τυπικών ημερομηνιών και χρόνων**

Οι ημερομηνίες εισάγονται σε μορφή 'YYYY-MM-DD', και οι ώρες ως TIME, διασφαλίζοντας συμβατότητα με MySQL και σωστό υπολογισμό (π.χ. διάρκειας, χρονικών επικαλύψεων).

### **Παράδειγμα:**

```
INSERT INTO Event (event_id, day_id, stage_id, start_time, end_time) VALUES (1, 1, 2,  
'18:00:00', '21:00:00');
```

### **5) Λογική διαχείριση ResaleQueue & Review**

Στη ResaleQueue, χρησιμοποιούνται λογικά status\_id, seller\_id, buyer\_id, request\_date.

Η Review εισάγεται μόνο για παραστάσεις που έχουν χρησιμοποιημένα εισιτήρια, στοιχείο που συνάδει με το InsertReview() stored procedure.

## **ΜΕΡΟΣ Β**

### **Αναφορά Εφαρμογής και Υλοποίησης Ερωτημάτων Βάσης Δεδομένων Μουσικού Φεστιβάλ**

#### **1. Αρχιτεκτονική Εφαρμογής και Τεχνική Υλοποίηση**

Η Εφαρμογή Βάσης Δεδομένων Μουσικού Φεστιβάλ έχει υλοποιηθεί ακολουθώντας μια δομημένη αρχιτεκτονική λογισμικού που δίνει έμφαση στη μοντελοποίηση, τη συντηρησιμότητα και το διαχωρισμό των ευθυνών.

#### **Αρχιτεκτονική MVC**

Η εφαρμογή ακολουθεί το πρότυπο Model-View-Controller (MVC):

**Μοντέλο (Model):** Υλοποιείται στο αρχείο database.py, το οποίο ενσωματώνει όλες τις λειτουργίες της βάσης δεδομένων, συμπεριλαμβανομένης της διαχείρισης σύνδεσης, εκτέλεσης ερωτημάτων και επεξεργασίας αποτελεσμάτων. Η κλάση Database παρέχει μια στιβαρή διεπαφή που αφαιρεί τις πολυπλοκότητες της βάσης δεδομένων από την υπόλοιπη εφαρμογή.

**Προβολή (View):** Η διεπαφή χρήστη υλοποιείται στο πακέτο ui, με το main\_window.py να παρέχει το κύριο παράθυρο εφαρμογής και το query\_frame.py να χειρίζεται την εμφάνιση αποτελεσμάτων και την οπτικοποίηση. Ο διαχωρισμός των στοιχείων της διεπαφής χρήστη σε ξεχωριστές μονάδες ενισχύει τη συντηρησιμότητα.

**Ελεγκτής (Controller):** Η λογική της εφαρμογής κατανέμεται μεταξύ του app.py (κύρια ροή εφαρμογής) και του queries.py (ορισμοί ερωτημάτων και μεταδεδομένα). Αυτός ο διαχωρισμός επιτρέπει την τροποποίηση των ορισμών των ερωτημάτων ανεξάρτητα από τη λογική της εφαρμογής.

#### **Αρθρωτή Σχεδίαση**

Η αρθρωτή σχεδίαση της εφαρμογής ενισχύει την οργάνωση του κώδικα και τη συντηρησιμότητα:

**Μονάδα Διαμόρφωσης:** Το αρχείο config.py απομονώνει όλες τις ρυθμίσεις διαμόρφωσης, διευκολύνοντας την προσαρμογή των παραμέτρων σύνδεσης βάσης δεδομένων και των ρυθμίσεων εφαρμογής χωρίς την τροποποίηση του βασικού κώδικα.

**Μονάδα Στυλ UI:** Το αρχείο styles.py συγκεντρώνει τους ορισμούς στυλ του UI, εξασφαλίζοντας συνεπή εμφάνιση σε όλη την εφαρμογή και απλοποιώντας μελλοντικές προσαρμογές στυλ.

**Αφαίρεση Βάσης Δεδομένων:** Η κλάση Database στο database.py παρέχει ένα ολοκληρωμένο επίπεδο αφαίρεσης που χειρίζεται τη διαχείριση σύνδεσης, εκτέλεση ερωτημάτων, παραμετροποίηση και χειρισμό σφαλμάτων.

**Οργάνωση Ερωτημάτων:** Το αρχείο queries.py δομεί και τα 15 απαιτούμενα ερωτήματα ως λεξικό με μεταδεδομένα σχετικά με παραμέτρους και περιγραφές, διευκολύνοντας την εύκολη πρόσβαση και διαχείριση.

### Χειρισμός Σφαλμάτων και Ανθεκτικότητα

Η εφαρμογή υλοποιεί στιβαρούς μηχανισμούς χειρισμού σφαλμάτων:

**Διαχείριση Σύνδεσης:** Η κλάση Database περιλαμβάνει μεθόδους για τον έλεγχο της κατάστασης σύνδεσης (is\_connected) και την αυτόματη επανασύνδεση όταν είναι απαραίτητο (reconnect\_if\_needed), ενισχύοντας την ανθεκτικότητα σε προβλήματα δικτύου.

**Παραμετροποιημένα Ερωτήματα:** Όλα τα ερωτήματα χρησιμοποιούν δέσμευση παραμέτρων για την αποφυγή επιθέσεων SQL injection, υποστηρίζοντας παράλληλα δυναμικές εισόδους ερωτημάτων.

**Ανατροφοδότηση Χρήστη:** Τα μηνύματα σφαλμάτων παρουσιάζονται στους χρήστες μέσω της γραφικής διεπαφής, παρέχοντας σαφείς πληροφορίες σχετικά με τα προβλήματα χωρίς να εκθέτουν ευαίσθητες τεχνικές λεπτομέρειες.

**Διαχείριση Συναλλαγών:** Οι κρίσμες λειτουργίες περιβάλλονται από συναλλαγές για τη διατήρηση της συνέπειας των δεδομένων ακόμα και όταν προκύπτουν σφάλματα.

## 2. Υλοποίηση και Βελτιστοποίηση Ερωτημάτων

### Οργάνωση Ερωτημάτων

Τα 15 απαιτούμενα ερωτήματα έχουν υλοποιηθεί με δομημένο τρόπο:

**Αποθήκευση Βάσει Λεξικού:** Όλα τα ερωτήματα αποθηκεύονται στο λεξικό QUERIES στο αρχείο queries.py, διευκολύνοντας την πρόσβαση σε αυτά μέσω αναγνωριστικού.

**Μεταδεδομένα:** Κάθε ερώτημα συνοδεύεται από μεταδεδομένα που περιλαμβάνουν ένα αναγνώσιμο όνομα, περιγραφή και ορισμούς παραμέτρων, ενισχύοντας τη χρηστικότητα της εφαρμογής.

**Ορισμοί Παραμέτρων:** Οι παράμετροι των ερωτημάτων ορίζονται με τύπους και περιγραφές, επιτρέποντας τη δυναμική δημιουργία διεπαφής χρήστη και τον κατάλληλο έλεγχο εγκυρότητας.

### Παραμετροποιημένα Ερωτήματα

Όλα τα ερωτήματα χρησιμοποιούν παραμετροποίηση για ασφάλεια και ευελιξία:

**Πρόληψη SQL Injection:** Οι παράμετροι περνούν ξεχωριστά από το κείμενο SQL και διαφεύγουν κατάλληλα από τον οδηγό της βάσης δεδομένων.

**Επικύρωση Τύπου:** Η εφαρμογή επικυρώνει τους τύπους παραμέτρων πριν από την εκτέλεση, διασφαλίζοντας την ακεραιότητα των δεδομένων.

**Δυναμική Είσοδος Χρήστη:** Η παραμετροποίηση επιτρέπει τη δυναμική είσοδο χρήστη χωρίς να διακυβεύεται η ασφάλεια ή να απαιτείται πολύπλοκος χειρισμός συμβολοσειρών SQL.

### Χαρακτηριστικά Βελτιστοποίησης Ερωτημάτων

Η εφαρμογή περιλαμβάνει προηγμένα χαρακτηριστικά βελτιστοποίησης για επιλεγμένα ερωτήματα:

**Εναλλακτικά Σχέδια Εκτέλεσης:** Τα ερωτήματα #4 (Μέσες Βαθμολογίες Καλλιτέχνη) και #6 (Βαθμολογίες Παραστάσεων Επισκέπτη) περιλαμβάνουν βελτιστοποιημένες εκδόσεις που χρησιμοποιούν υποδείξεις FORCE INDEX για να καθοδηγήσουν τον βελτιστοποιητή ερωτημάτων.

**Υποδείξεις INDEX:** Τα βελτιστοποιημένα ερωτήματα χρησιμοποιούν οδηγίες FORCE INDEX για να δώσουν προτεραιότητα σε συγκεκριμένα ευρετήρια:

-- Παράδειγμα από το Ερώτημα #4

FROM

Artist a FORCE INDEX (PRIMARY)

JOIN

Performance p FORCE INDEX (idx\_performance\_artist) ON a.artist\_id = p.artist\_id

JOIN

Review r FORCE INDEX (idx\_review\_performance) ON p.performance\_id = r.performance\_id

**Στατιστική Ανάλυση Απόδοσης:** Πολλαπλές επαναλήψεις εκτέλεσης επιτρέπουν τη στατιστική ανάλυση της απόδοσης των ερωτημάτων, λαμβάνοντας υπόψη τη μεταβλητότητα του συστήματος και τα αποτελέσματα της προσωρινής μνήμης.

**Ολοκληρωμένα Μετρικά:** Η εφαρμογή καταγράφει λεπτομερή μετρικά απόδοσης, συμπεριλαμβανομένων των ελάχιστων, μέγιστων, μέσων, διάμεσων και τυπικών αποκλίσεων των χρόνων εκτέλεσης.

### 3. Διεπαφή Χρήστη και Δυνατότητες Οπτικοποίησης

#### Πλαίσιο Διεπαφής Χρήστη

Η εφαρμογή διαθέτει μια ολοκληρωμένη διεπαφή χρήστη που έχει δημιουργηθεί με το Tkinter:

**Προσαρμοσμένο Σύστημα Στυλ:** Η μονάδα styles.py ορίζει συνεπή στυλ για όλα τα στοιχεία διεπαφής χρήστη, δημιουργώντας μια επαγγελματική και συνεκτική εμφάνιση.

**Αποκρίσιμη Διάταξη:** Η εφαρμογή χρησιμοποιεί διαχειριστές γεωμετρίας grid και pack με κατάλληλη στάθμιση για τη δημιουργία αποκρίσιμης διάταξης που προσαρμόζεται στην αλλαγή μεγέθους του παραθύρου.

**Διεπαφή με Καρτέλες:** Οι σύνθετες οπτικοποιήσεις και συγκρίσεις παρουσιάζονται σε μια διεπαφή με καρτέλες, επιτρέποντας στους χρήστες να περιηγηθούν μεταξύ διαφορετικών προβολών των δεδομένων.

**Γραμμή Κατάστασης:** Μια γραμμή κατάστασης παρέχει συνεχή ανατροφοδότηση σχετικά με την κατάσταση της εφαρμογής και τα αποτελέσματα των λειτουργιών.

#### Δυναμικός Χειρισμός Παραμέτρων

Η διεπαφή προσαρμόζεται δυναμικά στις απαρτήσεις των ερωτημάτων:

**Δημιουργία Φόρμας Παραμέτρων:** Για ερωτήματα με παραμέτρους, το UI δημιουργεί δυναμικά τα κατάλληλα πεδία εισόδου βάσει των μεταδεδομένων παραμέτρων.

**Επικύρωση Τύπου:** Οι τιμές των παραμέτρων επικυρώνονται σύμφωνα με τους καθορισμένους τύπους τους (ακέραιος, συμβολοσειρά, ημερομηνία) πριν από την εκτέλεση του ερωτήματος.

**Επεξηγηματικά Κείμενα:** Τα πεδία παραμέτρων περιλαμβάνουν επεξηγηματικά κείμενα με περιγραφές, βελτιώνοντας τη χρηστικότητα.

**Βοήθεια Περιεχομένου:** Για σύνθετες παραμέτρους, εμφανίζονται πρόσθετες πληροφορίες βοήθειας για την καθοδήγηση των χρηστών.

### **Χαρακτηριστικά Οπτικοποίησης**

Η εφαρμογή παρέχει εξελιγμένες δυνατότητες οπτικοποίησης δεδομένων:

**Αυτόματη Επιλογή Γραφήματος:** Η εφαρμογή αναλύει τη δομή των δεδομένων αποτελέσματος και επιλέγει αυτόματα τους κατάλληλους τύπους οπτικοποίησης:

Ραβδογράμματα για κατηγορικά έναντι αριθμητικών δεδομένων και ανάλυση κατανομής.

Γραμμικά γραφήματα για χρονοσειρές ή ακολουθιακά δεδομένα.

Γράφημα πίτας διαγράμματα για ανάλυση αναλογιών.

**Προσαρμογή Βάσει Δεδομένων:** Οι τίτλοι, οι ετικέτες και οι μορφές των γραφημάτων δημιουργούνται αυτόματα με βάση τα δεδομένα για την παροχή πλαισίου.

**Διαδραστικά Στοιχεία:** Τα γραφήματα περιλαμβάνουν διαδραστικά στοιχεία όπως επεξηγηματικά κείμενα και ετικέτες δεδομένων για βελτιωμένη εξερεύνηση δεδομένων.

**Πολλαπλές Επιλογές Οπτικοποίησης:** Για κάθε αποτέλεσμα ερωτήματος, οι χρήστες μπορούν να εξερευνήσουν διαφορετικούς τύπους οπτικοποίησης μέσω διεπαφής με καρτέλες.

### **Δυνατότητες Εξαγωγής**

Τα αποτελέσματα μπορούν να εξαχθούν για περαιτέρω ανάλυση:

**Εξαγωγή CSV:** Τα αποτελέσματα των ερωτημάτων μπορούν να εξαχθούν σε μορφή CSV για χρήση σε εφαρμογές υπολογιστικών φύλλων ή άλλα εργαλεία ανάλυσης.

**Εξαγωγή Excel:** Η εφαρμογή υποστηρίζει απευθείας εξαγωγή σε μορφή Excel, διατηρώντας τους τύπους δεδομένων και τα ονόματα στηλών.

**Διάλογος Εξαγωγής:** Ένας φιλικός προς το χρήστη διάλογος εξαγωγής επιτρέπει στους χρήστες να καθορίσουν τη θέση και τη μορφή εξαγωγής.

#### 4. Ανάλυση Απόδοσης και Τεχνικές Γνώσεις

##### Δοκιμή Πολλαπλών Επαναλήψεων

Η εφαρμογή υλοποιεί μια εξελιγμένη προσέγγιση ανάλυσης απόδοσης:

**Επαναλαμβανόμενη Εκτέλεση:** Η λειτουργία Σύγκρισης Σχεδίων Ερωτημάτων εκτελεί κάθε ερώτημα πολλές φορές για να λάβει υπόψη τη μεταβλητότητα του συστήματος.

**Προθέρμανση Cache:** Οι αρχικές εκτελέσεις επιτρέπουν τη θέρμανση της προσωρινής μνήμης της βάσης δεδομένων, με τις επόμενες εκτελέσεις να παρέχουν πιο συνεπείς μετρήσεις χρόνου.

**Στατιστική Συγκέντρωση:** Τα αποτελέσματα από πολλαπλές επαναλήψεις συγκεντρώνονται για την παροχή πιο αξιόπιστων μετρικών απόδοσης από τις μεμονωμένες εκτελέσεις.

##### Οπτικοποίηση με Box Plot

Τα δεδομένα απόδοσης οπτικοποιούνται με τη χρήση box plots:

**Οπτικοποίηση Κατανομής:** Τα box plots δείχνουν την κατανομή των χρόνων εκτέλεσης, τονίζοντας τη διάμεσο, τα τεταρτημόρια και τις ακραίες τιμές.

**Σύγκριση Δίπλα-Δίπλα:** Τα κανονικά και βελτιστοποιημένα ερωτήματα εμφανίζονται δίπλα-δίπλα για εύκολη οπτική σύγκριση.

**Στατιστικές Επισημάνσεις:** Η οπτικοποίηση περιλαμβάνει επισημάνσεις κειμένου με βασικά στατιστικά μετρικά για ολοκληρωμένη ανάλυση.

##### Ανάλυση Σχεδίου Ερωτήματος

Η εφαρμογή παρέχει βαθιές γνώσεις σχετικά με την εκτέλεση ερωτημάτων:

**Ανάκτηση Ίχνους Εκτέλεσης:** Για υποστηριζόμενα ερωτήματα, η εφαρμογή ανακτά λεπτομερή ίχνη εκτέλεσης χρησιμοποιώντας τη λειτουργικότητα optimizer\_trace της MariaDB.

**Σύγκριση Σχεδίων:** Οι χρήστες μπορούν να συγκρίνουν τα σχέδια εκτέλεσης μεταξύ κανονικών και βελτιστοποιημένων ερωτημάτων για να κατανοήσουν τα αποτελέσματα της βελτιστοποίησης.

**Μορφοποιημένη Εμφάνιση:** Τα ίχνη εκτέλεσης μορφοποιούνται και εμφανίζονται με αναγνώσιμο τρόπο με κατάλληλη εσοχή και δομή.

## Τεχνικές Γνώσεις

Η ανάπτυξη της εφαρμογής αποκάλυψε αρκετές σημαντικές τεχνικές γνώσεις:

**Μεταβλητότητα Απόδοσης Ερωτημάτων:** Η δοκιμή απόδοσης μίας εκτέλεσης αποδείχθηκε αναξιόπιστη λόγω επιδράσεων προσωρινής μνήμης και μεταβλητότητας φορτίου συστήματος.

**Πλαίσιο Αποτελεσματικότητας Ευρετηρίου:** Οι εξαναγκασμένες υποδείξεις ευρετηρίου μερικές φορές εμπόδιζαν αντί να βοηθούσαν την απόδοση ανάλογα με την κατανομή δεδομένων και τα μοτίβα ερωτημάτων.

**Αξία Στατιστικής Ανάλυσης:** Πολλαπλές επαναλήψεις με στατιστική ανάλυση παρείχαν πολύ πιο αξιόπιστα μετρικά απόδοσης από τις μεμονωμένες μετρήσεις.

**Σημασία Οπτικοποίησης:** Η μετατροπή πολύπλοκων σχεδίων ερωτημάτων και δεδομένων απόδοσης σε οπτικές μορφές ενίσχυσε σημαντικά την κατανόηση και την ανάλυση.

## 5. Προκλήσεις και Λύσεις

### Διαχείριση Σύνδεσης Βάσης Δεδομένων

**Πρόκληση:** Διατήρηση αξιόπιστων συνδέσεων βάσης δεδομένων καθ' όλη τη διάρκεια ζωής της εφαρμογής, ειδικά κατά τη διάρκεια περιόδων αδράνειας.

**Λύση:** Υλοποιήθηκε η μέθοδος `reconnect_if_needed` στην κλάση `Database` που ελέγχει την κατάσταση σύνδεσης πριν από κάθε εκτέλεση ερωτήματος και επανεγκαθιστά αυτόματα τις συνδέσεις όταν είναι απαραίτητο.

### Δυναμική Δημιουργία Διεπαφής Χρήστη

**Πρόκληση:** Δημιουργία ευέλικτης διεπαφής χρήστη που προσαρμόζεται σε διαφορετικά ερωτήματα με ποικίλες παραμέτρους.

**Λύση:** Αναπτύχθηκε μια προσέγγιση βασισμένη σε μεταδεδομένα όπου οι παράμετροι των ερωτημάτων ορίζονται στο αρχείο `queries.py` με πληροφορίες τύπου, επιτρέποντας στο UI να δημιουργεί δυναμικά κατάλληλα στοιχεία ελέγχου εισόδου και λογική επικύρωσης.

### Επιλογή Τύπου Γραφήματος

**Πρόκληση:** Αυτόματη επιλογή της καταλληλότερης οπτικοποίησης για διαφορετικές δομές δεδομένων.

**Λύση:** Υλοποιήθηκε ένας αλγόριθμος που αναλύει τα χαρακτηριστικά των δεδομένων αποτελέσματος (τύποι στηλών, αριθμός γραμμών, μοναδικές τιμές) για να καθορίσει τον καταλληλότερο τύπο οπτικοποίησης, με εναλλακτικές επιλογές όταν η κύρια οπτικοποίηση δεν είναι ιδανική.

### **Πολυπλοκότητα Ανάλυσης Απόδοσης**

**Πρόκληση:** Παροχή ουσιαστικών συγκρίσεων απόδοσης που λαμβάνουν υπόψη τη μεταβλητότητα του συστήματος.

**Λύση:** Αναπτύχθηκε μια στατιστική προσέγγιση που εκτελεί ερωτήματα πολλές φορές, υπολογίζει βασικά μετρικά (μέσος όρος, διάμεσος, ελάχιστο, μέγιστο, τυπική απόκλιση) και οπτικοποιεί την κατανομή χρησιμοποιώντας box plots για πιο οξιόπιστη σύγκριση.

### **Ερμηνεία Σχεδίου Ερωτήματος**

**Πρόκληση:** Η παροχή προσβάσιμων και κατανοητών δεδομένων σύνθετων σχεδίων εκτέλεσης στους χρήστες.

**Λύση:** Δημιουργήθηκε μια ειδική διεπαφή για σύγκριση σχεδίων με μορφοποιημένη εμφάνιση, σύγκριση δίπλα-δίπλα και επισήμανση βασικών μετρικών για να βοηθήσει τους χρήστες να κατανοήσουν τις διαφορές μεταξύ στρατηγικών εκτέλεσης ερωτημάτων.

## **6. Ειδική Εστίαση στις Υλοποιήσεις Ερωτημάτων**

### **Ερώτημα #1: Έσοδα Φεστιβάλ ανά Έτος και Μέθοδο Πληρωμής**

**Σκοπός:** Αναλύει τα έσοδα από πωλήσεις εισιτηρίων σε διαφορετικά έτη παρέχοντας ανάλυση ανά είδος πληρωμής.

**Τεχνική Υλοποίηση:** Χρησιμοποιεί GROUP BY με πολλαπλές στήλες και συναρτήσεις συγκέντρωσης (SUM, COUNT) για τη σύνοψη οικονομικών δεδομένων.

**Προσέγγιση Οπτικοποίησης:** Εμφανίζεται ως πινακοειδής αναφορά με προαιρετικά γραφήματα όπως ραβδόγραμμα και γραμμικό διάγραμμα τα οποία δείχνουν τα συνολικά έσοδα ανά έτος και γράφημα πίττας το οποίο δείχνει τα συνολικά έσοδα σε σχέση με τον τρόπο πληρωμής.

## Αποτελέσματα:

**Music Festival Database App**

**Select Query**

1. Festival Revenue by Year and Payment Method
2. Artists by Genre with Festival Participation
3. Artists with Multiple Warm-up Performances
4. Artist Average Ratings
5. Young Artists with Most Festival Participations
6. Visitor Attended Performances and Ratings
7. Festival with Lowest Technical Staff Experience
8. Unscheduled Support Staff for a Date
9. Visitors with Same Performance Attendance Count
10. Top 3 Genre Pairs in Artists

**Query Parameters**

No parameters required for this query.

**Query Details**

Shows the total revenue from ticket sales by year and payment method.

```

SELECT
    f.year,
    pm.name AS payment_method,
    SUM(t.price) AS total_revenue,
    COUNT(t.ticket_id) AS tickets_sold
FROM
    Ticket t
JOIN
    Event e ON t.event_id = e.event_id
JOIN
    festivalDay fd ON e.day_id = fd.day_id
JOIN
    Festival f ON fd.festival_id = f.festival_id

```

**Query Results**

Query: 1. Festival Revenue by Year and Payment Method | Time: 0.0777 seconds

year	payment_method	total_revenue	tickets_sold
2026	Credit Card	970.00	7
2026	Debit Card	540.00	5
2026	Bank Transfer	270.00	3
2025	Credit Card	910.00	8
2025	Debit Card	510.00	4
2025	Bank Transfer	255.00	3
2024	Credit Card	810.00	7
2024	Debit Card	450.00	5
2024	Bank Transfer	225.00	3
2023	Credit Card	1320.00	18
2023	Debit Card	480.00	6
2023	Bank Transfer	240.00	4
2022	Credit Card	2225.00	25
2022	Debit Card	640.00	6
2022	Bank Transfer	320.00	4
2021	Credit Card	660.00	10
2021	Debit Card	440.00	6
2021	Bank Transfer	220.00	4
2020	Credit Card	780.00	10
2020	Debit Card	520.00	6
2020	Bank Transfer	260.00	4
2019	Credit Card	1100.00	10
2019	Debit Card	560.00	6
2019	Bank Transfer	280.00	4
2018	Credit Card	720.00	10

**Data Visualization**

**Total Revenue by Year**

**Revenue Distribution by Payment Method**

**Revenue Trend by Year**

## Ερώτημα #2: Καλλιτέχνες ανά Είδος με Συμμετοχή σε Φεστιβάλ

**Σκοπός:** Παραθέτει καλλιτέχνες που ανήκουν σε συγκεκριμένο είδος με ένδειξη συμμετοχής σε φεστιβάλ για ένα συγκεκριμένο έτος.

**Τεχνική Υλοποίηση:** Χρησιμοποιεί μια έκφραση CASE με υποερώτημα για τον καθορισμό της κατάστασης συμμετοχής χωρίς να απαιτείται λειτουργία τροποποίησης δεδομένων.

**Θέματα Απόδοσης:** Χρησιμοποιεί το ενρετήριο idx\_artist\_genre για αποτελεσματικό φιλτράρισμα ανά είδος.

**Προσέγγιση Οπτικοποίησης:** Εμφανίζεται ως πινακοειδής αναφορά με το id, το όνομα, το είδος και υποείδος μουσικής των καλλιτεχνών και εάν συμμετείχαν ή όχι στο φεστιβάλ εκείνου του έτους.

### Αποτελέσματα:

artist_id	name	pseudonym	genre	subgenre	participated
2	Anna Vissi		Pop	Dance Pop	No
34	Christos Mastoras	Mastoras	Pop	Pop Rock	No
14	Despina Malea	Vandi	Pop	Dance Laiko	No
25	Elena Paparizou	Paparizou	Pop	Dance Pop	No
27	Ertelia Furraj	Eleni Foureira	Pop	Dance Pop	No
13	Eviidiki Theokleous	Eviidiki	Pop	Rock Pop	Yes
38	Giorgos Kakossaios		Pop	Laiko	No
11	Kati Garbi		Pop	Laiko Pop	Yes
23	Thodoris Feris	Feris	Pop	Pop Rock	No
29	Thodoris Marantinis		Pop	Pop Rock	No

Query executed successfully. 10 rows returned. Execution time: 0.0202 seconds

Execute Query   Execute With Trace   Compare Query Plans   Export Current Query   Compare Join Strategies

## Ερώτημα #3: Καλλιτέχνες με Πολλαπλές Εμφανίσεις Προθέρμανσης

**Σκοπός:** Εντοπίζει καλλιτέχνες που εμφανίστηκαν ως καλλιτέχνες προθέρμανσης περισσότερες από δύο φορές στο ίδιο φεστιβάλ.

**Τεχνική Υλοποίηση:** Χρησιμοποιεί GROUP BY με τη ρήτρα HAVING για φιλτράρισμα με βάση τα αποτελέσματα συγκέντρωσης.

**Προσέγγιση Οπτικοποίησης:** Εμφανίζεται ως πινακοειδής αναφορά με προαιρετικά γραφήματα ραβδόγραμμα και γράφημα πίττας που δείχνουν τον αριθμό προθερμάνσεων ανά καλλιτέχνη.

## Αποτελέσματα:

Music Festival Database App

Select Query

1. Festival Revenue by Year and Payment Method  
2. Artists by Genre with Festival Participation  
3. Artists with Multiple Warm-up Performances  
4. Artist Average Ratings  
5. Young Artists with Most Festival Participations  
6. Visitor Attended Performances and Ratings  
7. Festival with Lowest Technical Staff Experience  
8. Unscheduled Support Staff for a Date  
9. Visitors with Same Performance Attendance Count  
10. Top 3 Genre Pairs in Artists

Query Results

Export Results Show Execution Trace Visualize Data Query: 3. Artists with Multiple Warm-up Performances | Time: 0.0248s

artist_id	name	festival_name	year	warmup_count
39	Anastasia-Dimitra	New York World Music Heritage	2025	3
29	Thodoris Marantinis	Athens Reunion Festival	2024	3
2	Anna Vissi	London Park Festival	2020	3

Query Parameters

No parameters required for this query.

Query Details

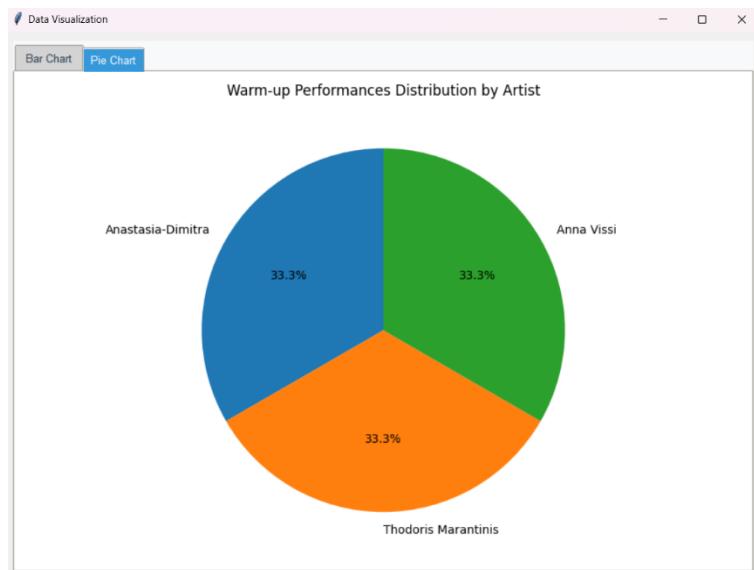
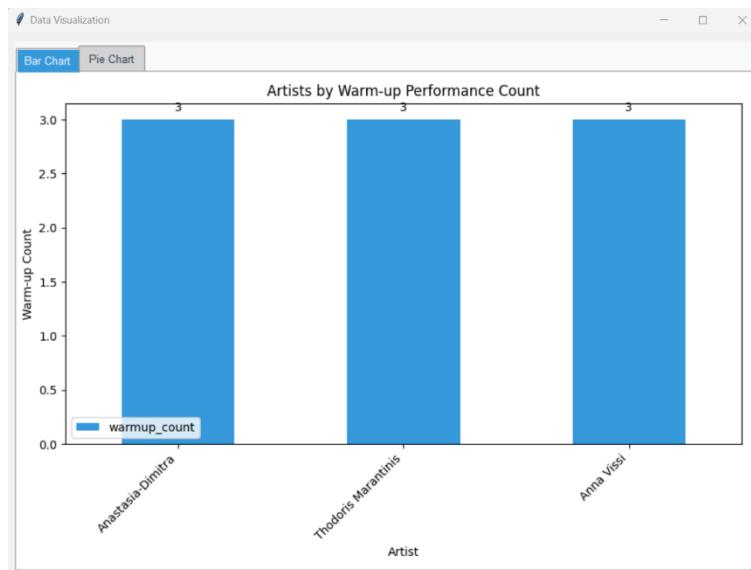
Finds artists who performed as warm-up more than 2 times in the same festival.

```

SELECT
    a.artist_id,
    a.name,
    f.name AS festival_name,
    f.year,
    COUNT(*) AS warmup_count
FROM
    Performance p
JOIN
    Artist a ON p.artist_id = a.artist_id
JOIN
    Event e ON p.event_id = e.event_id
JOIN
    Festival f ON e.festival_id = f.id
WHERE
    COUNT(*) > 2
    
```

Query executed successfully. 3 rows returned. Execution time: 0.0248 seconds

Execute Query Execute With Trace Compare Query Plans Export Current Query Compare Join Strategies



## Ερώτημα #4: Μέσες Βαθμολογίες Καλλιτέχνη

**Σκοπός:** Παρέχει μέσες βαθμολογίες παράστασης για συγκεκριμένους καλλιτέχνες σε διαφορετικές διαστάσεις.

**Τεχνική Υλοποίηση:** Υλοποιήθηκε τόσο σε κανονική όσο και σε βελτιστοποιημένη έκδοση για να δείξει τον αντίκτυπο της χρήσης ευρετηρίου.

**Θέματα Απόδοσης:** Η βελτιστοποιημένη έκδοση χρησιμοποιεί υποδείξεις FORCE INDEX για να κατευθύνει τον βελτιστοποιητή ερωτημάτων σε συγκεκριμένα ευρετήρια:

FROM

Artist a FORCE INDEX (PRIMARY)

JOIN

Performance p FORCE INDEX (idx\_performance\_artist) ON a.artist\_id = p.artist\_id

JOIN

Review r FORCE INDEX (idx\_review\_performance) ON p.performance\_id =

r.performance\_id

**Προσέγγιση Οπτικοποίησης:** Εμφανίζεται ως πινακοειδής που δείχνει τον μέσο όρο της αξιολόγησης του καλλιτέχνη ανά παράσταση και συνολικά, και τον συνολικό αριθμό αξιολογήσεων που έλαβε ο καλλιτέχνης.

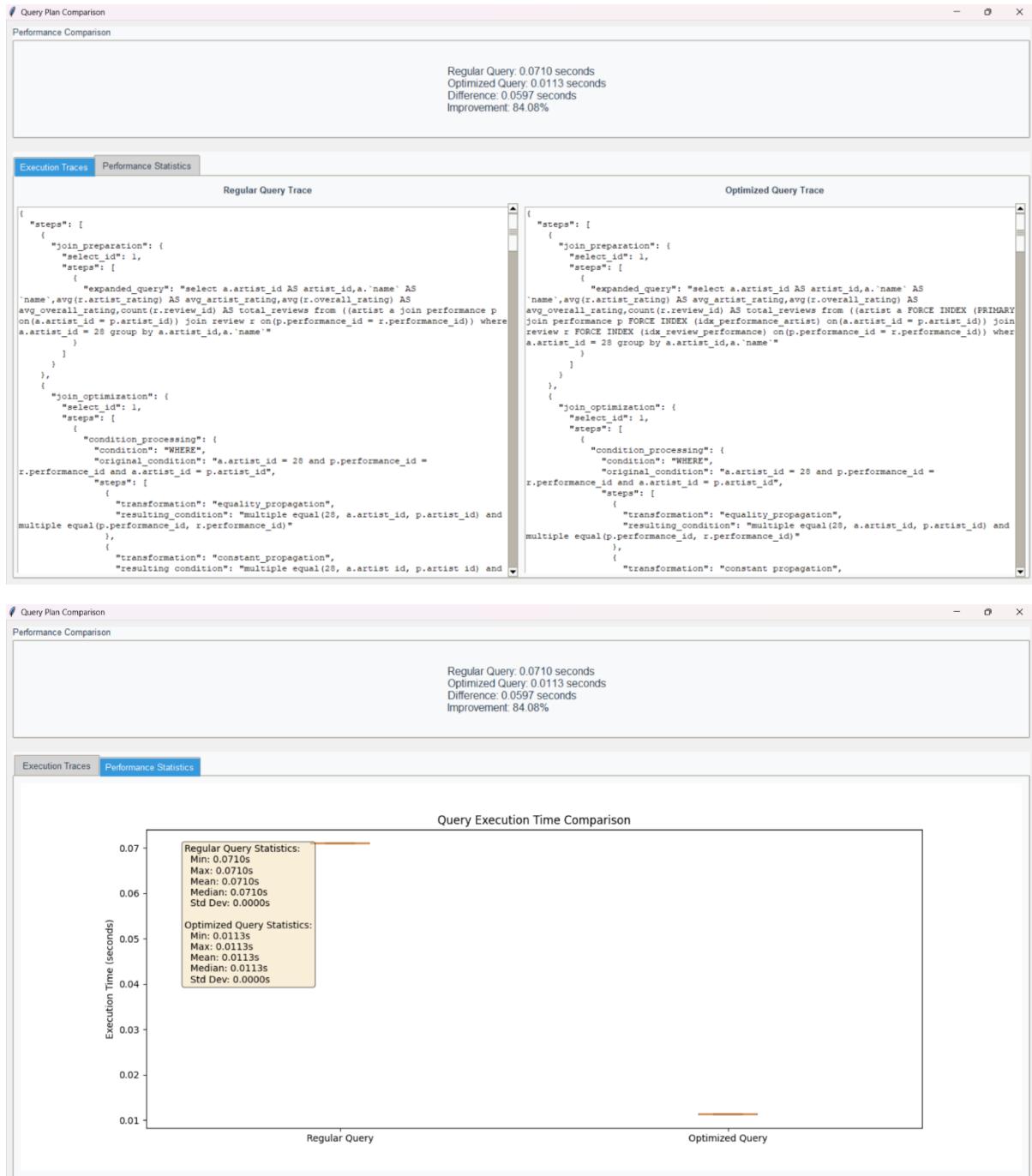
## Αποτελέσματα:

The screenshot shows a database query interface with the following sections:

- Select Query:** A list of 10 numbered options, with option 4 selected: "Artist Average Ratings".
- Query Parameters:** A field showing "artist\_id: 28".
- Query Details:** A description: "Shows the average ratings for an artist (performance and overall impression)."
- Query Editor:** The SQL query:

```
SELECT
    a.artist_id,
    a.name,
    AVG(r.artist_rating) AS avg_artist_rating,
    AVG(r.overall_rating) AS avg_overall_rating,
    COUNT(r.review_id) AS total_reviews
FROM
    Artist a
JOIN
    Performance p ON a.artist_id = p.artist_id
JOIN
    Review r ON p.performance_id = r.performance_id
WHERE
    a.artist_id = 28;
```
- Query Results:** A table titled "Query Results" showing one row for artist\_id 28, name Mariza Rizou, avg\_artist\_rating 5.0000, avg\_overall\_rating 5.0000, and total\_reviews 12.
- Bottom Bar:** Buttons for "Execute Query", "Execute With Trace", "Compare Query Plans", "Export Current Query", and "Compare Join Strategies".

## Σύγκριση με FORCE INDEX:



Για τα αποτελέσματα που εμφανίζονται πιο πάνω χρησιμοποιήθηκε ο καλλιτέχνης με artist\_id=28. Πιο κάτω δίνεται ο κώδικας που χρησιμοποιήθηκε για την εξαγωγή του ερωτήματος (για το συγκεκριμένο παράδειγμα καλλιτέχνη):

Κανονικό Ερώτημα:

SELECT

a.artist\_id,

```
a.name,  
AVG(r.artist_rating) AS avg_artist_rating,  
AVG(r.overall_rating) AS avg_overall_rating,  
COUNT(r.review_id) AS total_reviews  
FROM  
Artist a  
JOIN  
Performance p ON a.artist_id = p.artist_id  
JOIN  
Review r ON p.performance_id = r.performance_id  
WHERE  
a.artist_id = 28  
GROUP BY  
a.artist_id, a.name;
```

Βελτιστοποιημένο Ερώτημα με FORCE INDEX:

```
SELECT  
a.artist_id,  
a.name,  
AVG(r.artist_rating) AS avg_artist_rating,  
AVG(r.overall_rating) AS avg_overall_rating,  
COUNT(r.review_id) AS total_reviews  
FROM  
Artist a FORCE INDEX (PRIMARY)  
JOIN  
Performance p FORCE INDEX (idx_performance_artist) ON a.artist_id = p.artist_id  
JOIN  
Review r FORCE INDEX (idx_review_performance) ON p.performance_id =  
r.performance_id  
WHERE  
a.artist_id = 28  
GROUP BY  
a.artist_id, a.name;
```

Από τα αποτελέσματα των μετρήσεων παρατηρούμε ότι:

Έκδοση	Χρόνος Εκτέλεσης
Κανονική	0.0710 seconds
Με FORCE INDEX	0.0113 seconds
Διαφορά	0.0597 seconds
Βελτίωση	84.08%

Όπως φαίνεται από το αποτέλεσμα η χρήση των ευρετηρίων βελτίωσε σημαντικά την απόδοση με μείωση του χρόνου εκτέλεσης κατά περίπου 84%.

Η ανάλυση των ιχνών εκτέλεσης των δύο μεθόδων αποκαλύπτει τις εξής σημαντικές διαφορές μεταξύ των δύο ερωτημάτων:

### 1. Στρατηγική Πρόσβασης στον Πίνακα Review

- Κανονικό ερώτημα:** Ο βελτιστοποιητής επιλέγει τύπο πρόσβασης "ALL" (πλήρης σάρωση) για τον πίνακα Review, αγνοώντας το διαθέσιμο ευρετήριο idx\_review\_performance.
- Βελτιστοποιημένο ερώτημα:** Χρησιμοποιεί υποχρεωτικά το ευρετήριο idx\_review\_performance μέσω της οδηγίας FORCE INDEX, εφαρμόζοντας τύπο πρόσβασης "ref".

### 2. Επιλογή Ευρετηρίου για τον Πίνακα Performance

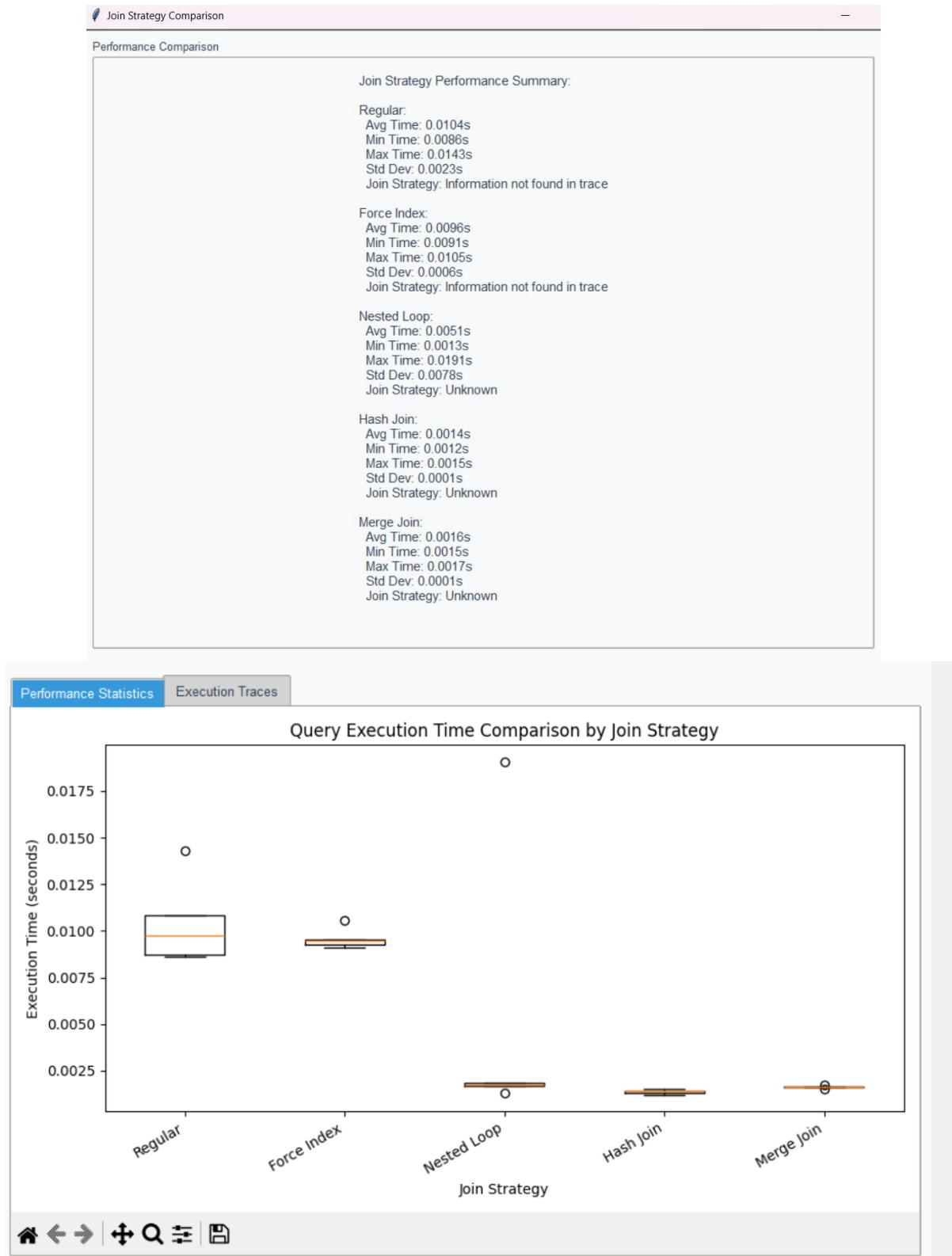
- Κανονικό ερώτημα:** Ο βελτιστοποιητής εξετάζει δύο πιθανά ευρετήρια ("possible\_keys": "idx\_performance\_artist,artist\_id") και επιλέγει το artist\_id, με κόστος ανάγνωσης.
- Βελτιστοποιημένο ερώτημα:** Χρησιμοποιεί αποκλειστικά το ευρετήριο idx\_performance\_artist ("forced\_index": true), το οποίο οδηγεί σε μικρότερο κόστος ανάγνωσης "read\_cost".

### 3. Στρατηγική Join και Τεχνικές Buffer

- Κανονικό ερώτημα:** Χρησιμοποιεί "Block Nested Loop" για τη σύνδεση του πίνακα Review, όπως υποδεικνύεται από το "using\_join\_buffer": "Block Nested Loop". Αυτή η στρατηγική απαιτεί προσωρινή αποθήκευση ενδιάμεσων αποτελεσμάτων στη μνήμη.

- Βελτιστοποιημένο ερώτημα:** Αποφεύγει τη χρήση join buffer, καθώς η χρήση του ευρετηρίου επιτρέπει πιο άμεση σύνδεση με τον πίνακα Performance. Δεν υπάρχει αναφορά στο "using\_join\_buffer" στο ίχνος, υποδεικνύοντας πιο αποδοτική σύνδεση.

### Συγκριτική Ανάλυση Στρατηγικών Join:



Βάσει των μετρήσεων που πραγματοποιήθηκαν, παρουσιάζουμε λεπτομερή ανάλυση της απόδοσης διαφορετικών στρατηγικών join για το ερώτημα #4, το οποίο υπολογίζει τη μέση βαθμολογία για έναν συγκεκριμένο καλλιτέχνη.

Στρατηγική Join	Μέσος χρόνος (ms)	Ελάχιστος (ms)	Μέγιστος (ms)	Τυπική απόκλιση (ms)	Βελτίωση
Regular	10.4	8.6	14.3	2.3	-
Force Index	9.6	9.1	10.5	0.6	7.7%
Nested Loop	5.1	1.3	19.1	7.8	51.0%
Hash Join	1.4	1.2	1.5	0.1	86.5%
Merge Join	1.6	1.5	1.7	0.1	84.6%

#### Ανάλυση Αποτελεσμάτων:

- Hash Join:** Αναδεικνύεται ως η βέλτιστη στρατηγική με μέσο χρόνο εκτέλεσης μόλις 1.4ms, προσφέροντας εντυπωσιακή βελτίωση 86.5% σε σχέση με το κανονικό ερώτημα. Επιπλέον, παρουσιάζει εξαιρετική σταθερότητα με πολύ μικρή τυπική απόκλιση (0.1ms), υποδεικνύοντας προβλέψιμη απόδοση σε κάθε εκτέλεση.
- Merge Join:** Παρόμοια απόδοση με το Hash Join, με μέσο χρόνο 1.6ms και βελτίωση 84.6%. Εμφανίζει επίσης εξαιρετική σταθερότητα με τυπική απόκλιση 0.1ms.
- Nested Loop:** Παρότι επιτυγχάνει σημαντική βελτίωση 51% στο μέσο χρόνο εκτέλεσης, παρουσιάζει τη μεγαλύτερη διακύμανση (τυπική απόκλιση 7.8ms) και τη μεγαλύτερη διαφορά μεταξύ ελάχιστου και μέγιστου χρόνου (από 1.3ms έως 19.1ms). Αυτό υποδηλώνει μη προβλέψιμη συμπεριφορά που μπορεί να επηρεάσει αρνητικά την εμπειρία χρήστη.
- Force Index:** Προσφέρει μια μέτρια βελτίωση 7.7% σε σχέση με το κανονικό ερώτημα, αλλά βελτιώνει σημαντικά τη σταθερότητα της απόδοσης, μειώνοντας την τυπική απόκλιση από 2.3ms σε 0.6ms.

#### Συμπεράσματα:

Οι μετρήσεις δείχνουν ξεκάθαρα ότι οι στρατηγικές Hash Join και Merge Join προσφέρουν κατά πολύ καλύτερη απόδοση για το ερώτημα #4, με το Hash Join να έχει ελαφρύ προβάδισμα. Η εντυπωσιακή βελτίωση σε συνδυασμό με την εξαιρετική σταθερότητα καθιστούν το Hash Join την προτεινόμενη επιλογή για την εκτέλεση του ερωτήματος #4.

Αξίζει να σημειωθεί ότι η απλή χρήση του FORCE INDEX προσφέρει μέτρια βελτίωση, γεγονός που υποδεικνύει ότι η επιλογή της κατάλληλης στρατηγικής join είναι πιο σημαντική από την επιβολή συγκεκριμένων ευρετηρίων για αυτό το ερώτημα.

Η μεγάλη διακύμανση στην απόδοση του Nested Loop σημαίνει ότι παρά τη σημαντική βελτίωση στο μέσο χρόνο, δεν αποτελεί αξιόπιστη επιλογή για περιβάλλοντα παραγωγής όπου η σταθερότητα είναι εξίσου σημαντική με την ταχύτητα.

### **Ερώτημα #5: Νέοι Καλλιτέχνες με τις Περισσότερες Συμμετοχές σε Φεστιβάλ**

**Σκοπός:** Εντοπίζει καλλιτέχνες κάτω των 30 ετών με τον υψηλότερο αριθμό συμμετοχών σε φεστιβάλ.

**Τεχνική Υλοποίηση:** Χρησιμοποιεί τη συνάρτηση TIMESTAMPDIFF για δυναμικό υπολογισμό ηλικιών κατά το χρόνο του ερωτήματος.

**Θέματα Απόδοσης:** Χρησιμοποιεί το ευρετήριο idx\_artist\_birthdate για αποτελεσματικό φιλτράρισμα ηλικίας.

**Προσέγγιση Οπτικοποίησης:** Εμφανίζεται ως πινακοειδής αναφορά με τους καλλιτέχνες κάτω των 30 ετών και πόσες συμμετοχές είχαν σε φεστιβάλ και ραβδόγραμμα και γράφημα πίττα που δείχνουν τον αριθμό των συμμετοχών των καλλιτεχνών στα φεστιβάλ.

### **Αποτελέσματα:**

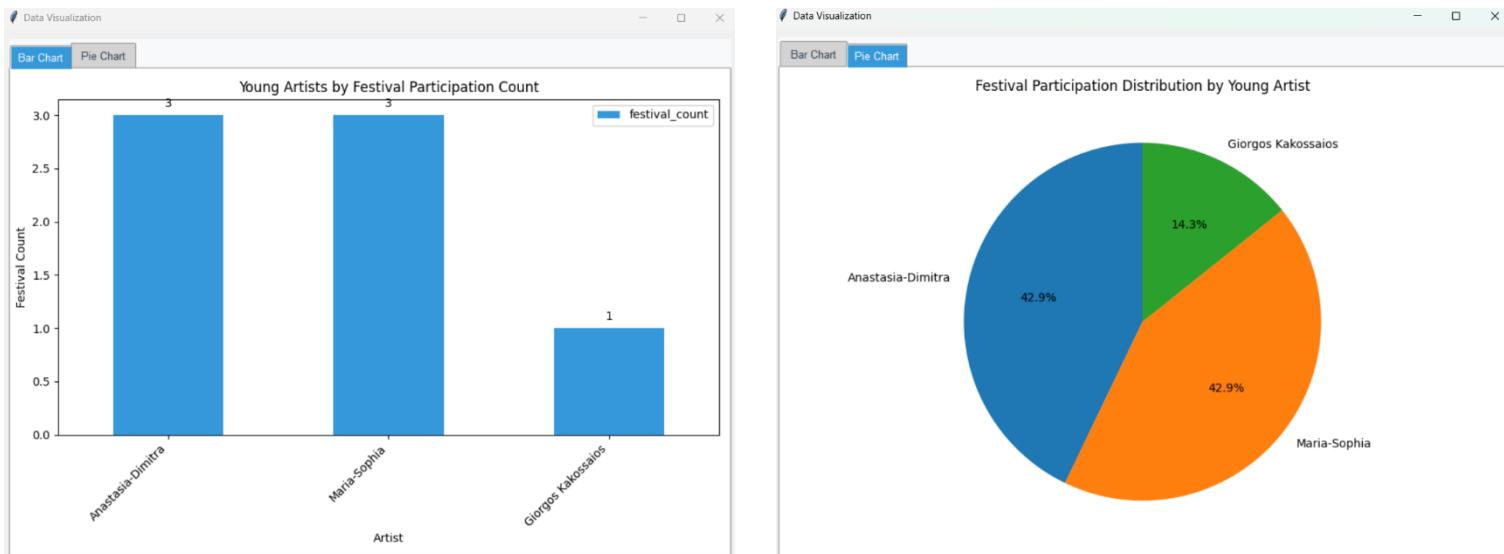
The screenshot shows the 'Music Festival Database App' interface. On the left, under 'Select Query', query 5 is selected: 'Young Artists with Most Festival Participations'. The 'Query Details' pane contains the explanatory text: 'Lists young artists (under 30) with the most festival participations.' Below it is the SQL query:

```
SELECT
    a.artist_id,
    a.name,
    TIMESTAMPDIFF(YEAR, a.birthdate, CURDATE()) AS age,
    COUNT(DISTINCT f.festival_id) AS festival_count
FROM
    Artist a
JOIN
    Performance p ON a.artist_id = p.artist_id
JOIN
    Event e ON p.event_id = e.event_id
JOIN
    FestivalDay fd ON e.day_id = fd.day_id
```

The 'Query Results' pane displays the output:

artist_id	name	age	festival_count
39	Anastasia-Dimitra	21	3
37	Maria-Sophia	24	3
38	Giorgos Kakossaios	25	1

At the bottom, the status bar indicates: 'Query executed successfully. 3 rows returned. Execution time: 0.0306 seconds.'



## Ερώτημα #6: Παραστάσεις που παρακολουθήθηκαν και Βαθμολογίες Επισκέπτη

**Σκοπός:** Δείχνει όλες τις παραστάσεις που παρακολούθησε ένας συγκεκριμένος επισκέπτης μαζί με το μέσο όρο των αξιολογήσεων του.

**Τεχνική Υλοποίηση:** Χρησιμοποιεί LEFT JOIN για να συμπεριλάβει παραστάσεις χωρίς κριτικές και μια έκφραση CASE για το χειρισμό διαφορετικών τύπων εκτελεστών (καλλιτέχνης έναντι συγκροτήματος).

**Θέματα Απόδοσης:** Η βελτιστοποιημένη έκδοση χρησιμοποιεί πολλαπλές υποδείξεις FORCE INDEX για τη βελτίωση της απόδοσης της σύνδεσης:

**FROM**

Ticket t FORCE INDEX (idx\_ticket\_visitor)

**JOIN**

Event e FORCE INDEX (PRIMARY) ON t.event\_id = e.event\_id

...

**LEFT JOIN**

Review r FORCE INDEX (idx\_review\_visitor) ON p.performance\_id = r.performance\_id

AND r.visitor\_id = t.visitor\_id

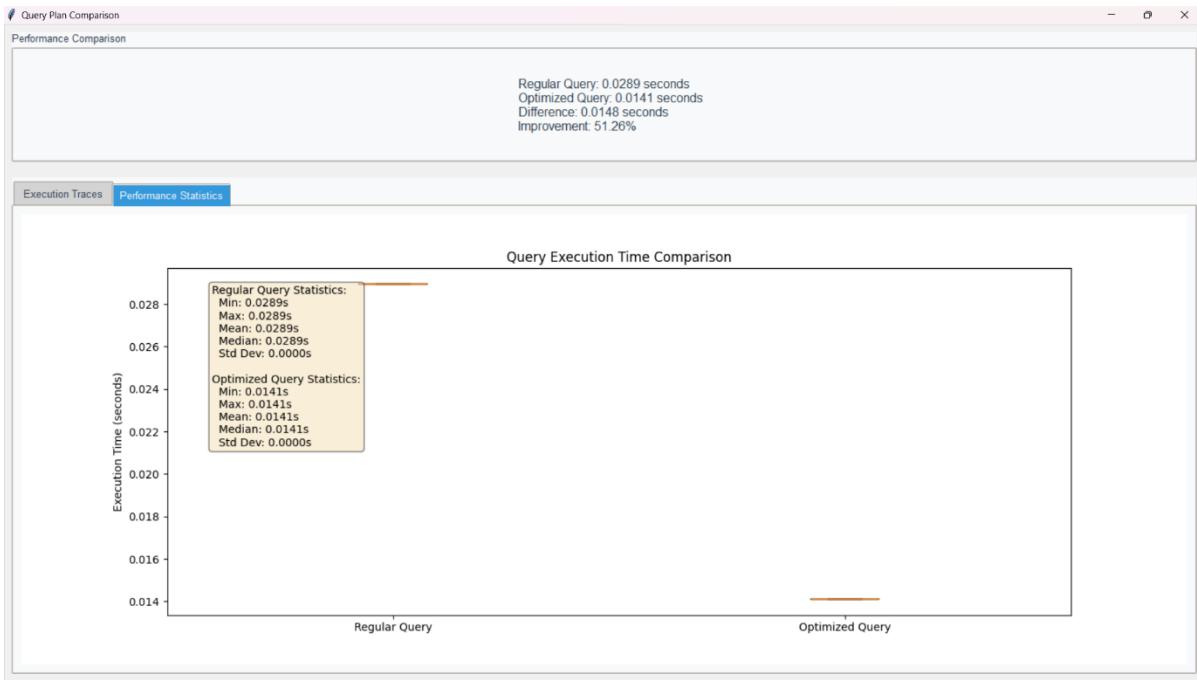
**Προσέγγιση Οπτικοποίησης:** Εμφανίζεται ως πινακοειδής αναφορά με τις παραστάσεις που παρακολούθησε ο επισκέπτης, πόσες αξιολογήσεις έκανε για κάθε παράσταση και τον μέσο όρο της κάθε αξιολόγησης που έκανε.

## Αποτελέσματα:

A screenshot of a database application interface. At the top, there's a navigation bar with tabs like 'Select', 'Query', 'Export Results', 'Show Execution Trace', 'Visualize Data', and others. Below the navigation is a 'Query Results' section showing a table with columns: event\_name, festival\_date, performer, performance\_type, avg\_rating, and review\_count. The table contains three rows for 'Rock Opening Night' on '2017-07-01'. The rows show performers Vasilis Papakonstantinou (Warm Up), Pyx Lax (Headline), and Sokratis Malamas (Special Guest) with average ratings of 3.0000, 4.0000, and 4.0000 respectively, and all having a review count of 1. Above the table, a message says '3 rows Query: 6. Visitor Attended Performances and Ratings | Time: 0.0099s'. To the left of the results, there's a 'Query Parameters' section with a 'visitor\_id' input set to '2'. Below that is a 'Query Details' section with a note: 'Shows performances attended by a specific visitor and their average ratings.' At the bottom, there's a code editor window showing the SQL query used: SELECT e.name AS event\_name, fd.festival\_date, CASE WHEN p.artist\_id IS NOT NULL THEN a.name WHEN p.band\_id IS NOT NULL THEN b.name ELSE 'Unknown' END AS performer, pt.name AS performance\_type, AVG(r.overall\_rating) AS avg\_rating, COUNT(r.review\_id) AS review\_count FROM ticket t ... followed by a note 'Query executed successfully. 3 rows returned. Execution time: 0.0099 seconds' and several execution options: 'Execute Query', 'Execute With Trace', 'Compare Query Plans', 'Export Current Query', and 'Compare Join Strategies'.

## Σύγκριση με FORCE INDEX:

A screenshot of a 'Query Plan Comparison' tool. It shows two main sections: 'Performance Comparison' and 'Execution Traces'. In the 'Performance Comparison' section, it says 'Regular Query: 0.0289 seconds', 'Optimized Query: 0.0141 seconds', 'Difference: 0.0148 seconds', and 'Improvement: 51.26%'. In the 'Execution Traces' section, there are two side-by-side traces. The left one is labeled 'Regular Query Trace' and the right one is labeled 'Optimized Query Trace'. Both traces are represented as JSON objects showing the steps of the query execution, including joins, conditions, and indexes used. The optimized trace shows more detailed information about the execution plan, including the use of force indexes and specific join orders.



Για τα αποτελέσματα που εμφανίζονται πιο πάνω χρησιμοποιήθηκε ο επισκέπτης με visitor\_id=2. Πιο κάτω δίνεται ο κώδικας που χρησιμοποιήθηκε για την εξαγωγή του ερωτήματος (για το συγκεκριμένο παράδειγμα καλλιτέχνη):

Κανονικό Ερώτημα:

SELECT

e.name AS event\_name,

fd.festival\_date,

CASE

WHEN p.artist\_id IS NOT NULL THEN a.name

WHEN p.band\_id IS NOT NULL THEN b.name

ELSE 'Unknown'

END AS performer,

pt.name AS performance\_type,

AVG(r.overall\_rating) AS avg\_rating,

COUNT(r.review\_id) AS review\_count

FROM

Ticket t

JOIN

Event e ON t.event\_id = e.event\_id

JOIN

FestivalDay fd ON e.day\_id = fd.day\_id

JOIN

Performance p ON e.event\_id = p.event\_id

LEFT JOIN

Artist a ON p.artist\_id = a.artist\_id

LEFT JOIN

Band b ON p.band\_id = b.band\_id

JOIN

PerformanceType pt ON p.type\_id = pt.type\_id

LEFT JOIN

Review r ON p.performance\_id = r.performance\_id AND r.visitor\_id = t.visitor\_id

WHERE

t.visitor\_id = 2

AND t.is\_active = FALSE

GROUP BY

e.name, fd.festival\_date, performer, pt.name

ORDER BY

fd.festival\_date DESC;

Βελτιστοποιημένο Ερώτημα με FORCE INDEX:

SELECT

e.name AS event\_name,

fd.festival\_date,

CASE

WHEN p.artist\_id IS NOT NULL THEN a.name

WHEN p.band\_id IS NOT NULL THEN b.name

ELSE 'Unknown'

END AS performer,

pt.name AS performance\_type,

AVG(r.overall\_rating) AS avg\_rating,

COUNT(r.review\_id) AS review\_count

FROM

Ticket t FORCE INDEX (idx\_ticket\_visitor)

JOIN

Event e FORCE INDEX (PRIMARY) ON t.event\_id = e.event\_id

JOIN

FestivalDay fd ON e.day\_id = fd.day\_id

JOIN

Performance p ON e.event\_id = p.event\_id

LEFT JOIN

Artist a ON p.artist\_id = a.artist\_id

LEFT JOIN

Band b ON p.band\_id = b.band\_id

JOIN

PerformanceType pt ON p.type\_id = pt.type\_id

LEFT JOIN

Review r FORCE INDEX (idx\_review\_visitor) ON p.performance\_id = r.performance\_id

AND r.visitor\_id = t.visitor\_id

WHERE

t.visitor\_id = 2

AND t.is\_active = FALSE

GROUP BY

e.name, fd.festival\_date, performer, pt.name

ORDER BY

fd.festival\_date DESC;

Από τα αποτελέσματα των μετρήσεων παρατηρούμε ότι:

Έκδοση	Χρόνος Εκτέλεσης
Κανονική	0.0289 seconds
Με FORCE INDEX	0.0141 seconds
Διαφορά	0.0148 seconds
Βελτίωση	51.26%

Όπως φαίνεται από το αποτέλεσμα η χρήση των ευρετηρίων βελτίωσε σημαντικά την απόδοση με μείωση του χρόνου εκτέλεσης κατά 51%.

Η ανάλυση των ιχνών εκτέλεσης των δύο μεθόδων αποκαλύπτει τις εξής σημαντικές διαφορές μεταξύ των δύο ερωτημάτων:

## 1. Στρατηγική Πρόσβασης στον Πίνακα Ticket (t)

- Κανονικό ερώτημα:** Εκτελεί πλήρη σάρωση του πίνακα Ticket και στη συνέχεια φιλτράρει τα αποτελέσματα με βάση το visitor\_id. Αυτό οδηγεί σε εξέταση περίπου 200+ γραμμών.
- Βελτιστοποιημένο ερώτημα:** Χρησιμοποιεί άμεσα το ευρετήριο idx\_ticket\_visitor μέσω της οδηγίας FORCE INDEX, πραγματοποιώντας αναζήτηση μόνο για τα εισιτήρια του συγκεκριμένου επισκέπτη. Εξετάζει μόλις 5-10 γραμμές.

## 2. Σειρά Προσπέλασης Πινάκων και Στρατηγική Join

- Κανονικό ερώτημα:** Ο βελτιστοποιητής επιλέγει να ξεκινήσει από τον πίνακα Event και στη συνέχεια να συνδεθεί με τον Ticket, ακολουθώντας μια λιγότερο αποδοτική σειρά για το συγκεκριμένο ερώτημα.
- Βελτιστοποιημένο ερώτημα:** Αλλάζει τη σειρά προσπέλασης, ξεκινώντας από τον πίνακα Ticket (με το φίλτρο visitor\_id), μειώνοντας σημαντικά το αρχικό σύνολο γραμμών για τις επόμενες συνδέσεις.

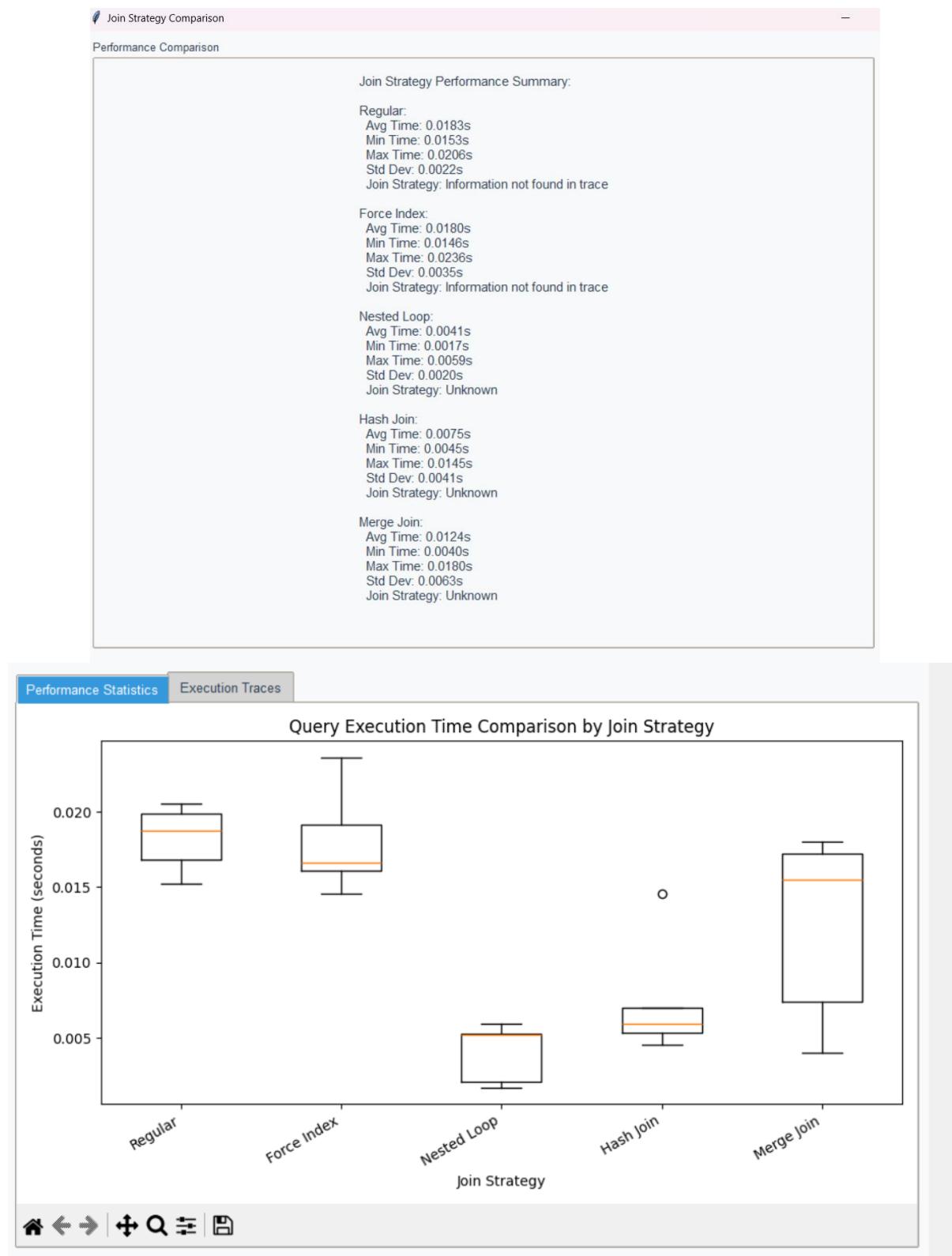
## 3. Διαχείριση της Σύνδεσης με τον Πίνακα Review

- Κανονικό ερώτημα:** Δεν αξιοποιεί κάποιο ευρετήριο για το φιλτράρισμα του πίνακα Review με βάση το visitor\_id, εκτελώντας έτσι πλήρη σάρωση για κάθε γραμμή αποτελέσματος από προηγούμενες συνδέσεις. Χρησιμοποιεί "Block Nested Loop" join που έχει υψηλότερο κόστος.
- Βελτιστοποιημένο ερώτημα:** Χρησιμοποιεί το ευρετήριο idx\_review\_visitor μέσω FORCE INDEX, επιτρέποντας πολύ πιο αποδοτικό φιλτράρισμα των αξιολογήσεων. Το κόστος της σύνδεσης μειώνεται σημαντικά και η μέθοδος join βελτιώνεται.

## 4. Τεχνικές Buffer και Προσωρινής Αποθήκευσης

- Κανονικό ερώτημα:** Παρατηρείται αυξημένη χρήση του buffer προσωρινής αποθήκευσης λόγω των μη αποδοτικών join (χρήση "using\_join\_buffer": "Block Nested Loop").
- Βελτιστοποιημένο ερώτημα:** Ελαχιστοποιεί την ανάγκη για join buffers καθώς η χρήση των ευρετηρίων επιτρέπει πιο άμεσες συνδέσεις.

## Συγκριτική Ανάλυση Στρατηγικών Join:



Βάσει των διαθέσιμων μετρήσεων, παρουσιάζουμε λεπτομερή ανάλυση της απόδοσης διαφορετικών στρατηγικών join για το ερώτημα #6, το οποίο αφορά την εύρεση των

παραστάσεων που έχει παρακολουθήσει ένας συγκεκριμένος επισκέπτης και των αντίστοιχων βαθμολογιών.

Στρατηγική Join	Μέσος χρόνος (ms)	Ελάχιστος (ms)	Μέγιστος (ms)	Τυπική απόκλιση (ms)	Βελτίωση
Regular	18.3	15.3	20.6	2.2	-
Force Index	18.0	14.6	23.6	3.5	1.6%
Nested Loop	4.1	1.7	5.9	2.0	77.6%
Hash Join	7.5	4.5	14.5	4.1	59.0%
Merge Join	12.4	4.0	18.0	6.3	32.2%

#### Ανάλυση Αποτελεσμάτων:

- Nested Loop:** Αποτελεί την πιο αποδοτική στρατηγική για το ερώτημα #6, με εντυπωσιακή βελτίωση 77.6% σε σχέση με το κανονικό ερώτημα. Παρουσιάζει συγκριτικά καλή σταθερότητα με τυπική απόκλιση 2.0ms και μικρό εύρος μεταξύ ελάχιστης και μέγιστης τιμής (1.7ms - 5.9ms).
- Hash Join:** Παρέχει σημαντική βελτίωση 59.0% με μέσο χρόνο εκτέλεσης 7.5ms, αλλά εμφανίζει μεγαλύτερη διακύμανση (τυπική απόκλιση 4.1ms) σε σχέση με το Nested Loop. Το εύρος τιμών από 4.5ms έως 14.5ms υποδηλώνει λιγότερο συνεπή απόδοση.
- Merge Join:** Προσφέρει μέτρια βελτίωση 32.2% με μέσο χρόνο 12.4ms, αλλά έχει τη μεγαλύτερη διακύμανση από όλες τις στρατηγικές (τυπική απόκλιση 6.3ms) και ευρύ διάστημα μεταξύ ελάχιστης και μέγιστης τιμής (4.0ms - 18.0ms).
- Force Index:** Παρέχει οριακή βελτίωση μόλις 1.6% σε σχέση με το κανονικό ερώτημα και μάλιστα με αυξημένη αστάθεια (τυπική απόκλιση 3.5ms έναντι 2.2ms του κανονικού ερωτήματος).

#### Συμπεράσματα:

Τα αποτελέσματα των μετρήσεων δείχνουν ότι για το ερώτημα #6, που περιλαμβάνει πολλαπλές συνδέσεις πινάκων με πολύπλοκες σχέσεις, η στρατηγική Nested Loop αποδεικνύεται πολύ πιο αποτελεσματική από τις υπόλοιπες. Αυτό αποτελεί ενδιαφέρον εύρημα, καθώς διαφέρει από τα αποτελέσματα του ερωτήματος #4, όπου το Hash Join ήταν η βέλτιστη επιλογή.

Η διαφορά στην επίδοση των στρατηγικών μεταξύ των δύο ερωτημάτων υπογραμμίζει πόσο σημαντικό είναι να αξιολογούμε τις στρατηγικές join με βάση τα συγκεκριμένα χαρακτηριστικά κάθε ερωτήματος και της δομής των δεδομένων. Στην περίπτωση του

ερωτήματος #6, το Nested Loop επιτυγχάνει αποδοτικότερο φίλτραρισμα των δεδομένων σε πρώιμο στάδιο της εκτέλεσης, μειώνοντας δραματικά τον όγκο των ενδιάμεσων αποτελεσμάτων.

Αξιοσημείωτο είναι επίσης ότι η απλή χρήση του FORCE INDEX παρέχει ελάχιστη βελτίωση για το ερώτημα #6, που υποδεικνύει ότι η επιλογή της κατάλληλης στρατηγικής join είναι πολύ πιο σημαντική από την υποχρεωτική χρήση συγκεκριμένων ευρετηρίων για αυτό το ερώτημα.

Με βάση τα παραπάνω, η προτεινόμενη προσέγγιση για το ερώτημα #6 είναι η χρήση της στρατηγικής Nested Loop, η οποία συνδυάζει εξαιρετική απόδοση με ικανοποιητική σταθερότητα.

## Ερώτημα #7: Φεστιβάλ με τη Χαμηλότερη Εμπειρία Τεχνικού Προσωπικού

**Σκοπός:** Εντοπίζει φεστιβάλ με τη χαμηλότερη μέσο επίπεδο εμπειρίας τεχνικού προσωπικού.

**Τεχνική Υλοποίηση:** Χρησιμοποιεί πολλαπλές συνδέσεις για να συνδέσει τις αναθέσεις προσωπικού με τα επίπεδα εμπειρίας και υπολογίζει τη μέση εμπειρία ως μετρική.

**Προσέγγιση Οπτικοποίησης:** Εμφανίζεται ως πινακοειδής αναφορά με το όνομα, το id, την χρονολογία του φεστιβάλ και τον μέσο όρο εμπειρίας του προσωπικού.

### Αποτελέσματα:

The screenshot shows the 'Music Festival Database App' interface. On the left, the 'Select Query' sidebar lists various queries numbered 1 to 10. Query 7 is selected, which is 'Festival with Lowest Technical Staff Experience'. Below this, the 'Query Parameters' section indicates 'No parameters required for this query'. The 'Query Details' section contains a brief description: 'Finds the festival with the lowest average experience level of technical staff.' The main area displays the 'Query Results' table. The table has four columns: 'festival\_id', 'festival\_name', 'year', and 'avg\_experience\_level'. A single row is shown for festival\_id 2, named 'New York Sound Experience', from the year 2018, with an average experience level of 4.0000. At the bottom, a message states 'Query executed successfully. 1 rows returned. Execution time: 0.0571 seconds' and a row of buttons includes 'Execute Query' (highlighted in green), 'Execute With Trace', 'Compare Query Plans', 'Export Current Query', and 'Compare Join Strategies'.

festival_id	festival_name	year	avg_experience_level
2	New York Sound Experience	2018	4.0000

## Ερώτημα #8: Μη Προγραμματισμένο Προσωπικό Υποστήριξης για μια Ημερομηνία

**Σκοπός:** Παραθέτει προσωπικό υποστήριξης που δεν έχει ανατεθεί σε εκδηλώσεις σε συγκεκριμένη ημερομηνία.

**Τεχνική Υλοποίηση:** Χρησιμοποιεί ένα υποερώτημα NOT IN με φίλτραρισμα ημερομηνίας για την εύρεση μη ανατεθειμένου προσωπικού.

**Θέματα Απόδοσης:** Χρησιμοποιεί το ευρετήριο idx\_staff\_role για αποτελεσματικό φίλτραρισμα ρόλων.

**Προσέγγιση Οπτικοποίησης:** Εμφανίζεται κυρίως ως πινακοειδής αναφορά που δείχνει την το id, το όνομα, την ηλικία και το επίπεδο εμπειρίας του διαθέσιμου προσωπικού.

### Αποτελέσματα:

The screenshot shows a database query interface with the following details:

- Query Results:** A table titled "Query Results" showing 68 rows of data. The columns are staff\_id, name, age, and experience\_level. The data includes names like Alexandra Michalidou, Andreas Georgiou, Athina Nikolau, etc., with ages ranging from 25 to 37 and experience levels from Beginner to Experienced.
- Query Parameters:** A field showing "date: 2019-08-11".
- Query Details:** A note stating "Lists support staff not scheduled for a specific date."
- SQL Query:**

```
SELECT
    s.staff_id,
    s.name,
    s.age,
    el.name AS experience_level
FROM
    Staff s
JOIN
    ExperienceLevel el ON s.level_id = el.level_id
JOIN
    StaffRole sr ON s.role_id = sr.role_id
WHERE
    sr.name = 'Support'
```
- Execution Summary:** "Query executed successfully, 68 rows returned. Execution time: 0.0410 seconds".
- Action Buttons:** Execute Query, Execute With Trace, Compare Query Plans, Export Current Query, Compare Join Strategies.

## Ερώτημα #9: Επισκέπτες με τον ίδιο Αριθμό Παρακολούθησης Παραστάσεων

**Σκοπός:** Εντοπίζει ομάδες επισκεπτών που παρακολούθησαν τον ίδιο αριθμό παραστάσεων σε ένα δεδομένο έτος.

**Τεχνική Υλοποίηση:** Χρησιμοποιεί ένθετα υποερωτήματα με τη ρήτρα HAVING για την εύρεση αντίστοιχων αριθμών παρακολούθησης μεταξύ διαφορετικών επισκεπτών.

**Προσέγγιση Οπτικοποίησης:** Εμφανίζεται ως πινακοειδής αναφορά με το id και το όνομα του κάθε επισκέπτη και πόσες παραστάσεις παρακολούθησε ο κάθε ένας, ποια χρονολογία και σε ποιο φεστιβάλ.

## Αποτελέσματα:

The screenshot shows a database application interface with a sidebar of queries and a main panel for 'Query Results'. The sidebar lists numbered queries from 4 to 13. The main panel displays the results of Query 9: 'Visitors with Same Performance Attendance Count'. The results table has columns: visitor\_id, visitor\_name, year, attendance\_count, and festivals\_attended. The data shows five visitors (31, 32, 33, 34, 35) attending 4, 4, 5, 5, and 5 performances respectively at Thessaloniki Jazz Festival, Thessaloniki Jazz Festival, Sydney Opera Sounds, Sydney Opera Sounds, and Sydney Opera Sounds.

visitor_id	visitor_name	year	attendance_count	festivals_attended
31	Dimitris Vasileiou	2023	4	Thessaloniki Jazz Festival
32	Nikos Savidis	2023	4	Thessaloniki Jazz Festival
33	Georgia Kostas	2022	5	Sydney Opera Sounds
34	Athina Ioannidis	2022	5	Sydney Opera Sounds
35	Andreas Ioannidis	2022	5	Sydney Opera Sounds

**Query Details:** Finds visitors who attended the same number of performances in a year (with more than 3 attendances).

```

SELECT
    v.visitor_id,
    CONCAT(v.first_name, ' ', v.last_name) AS visitor_name,
    fd.year,
    COUNT(fd.attendance_count) AS attendance_count,
    GROUP_CONCAT(DISTINCT f.name) AS festivals_attended
FROM
    Visitor v
JOIN
    fd Get attendance counts by visitor and year
SELECT
    visitor_id,
    YEAR(fd.festival_date) AS year,
    COUNT(fd.attendance_count) AS attendance_count
    GROUP BY visitor_id, year
    ORDER BY attendance_count DESC
    LIMIT 1;
  
```

Query executed successfully: 5 rows returned. Execution time: 0.1023 seconds

Buttons: Execute Query, Execute With Trace, Compare Query Plans, Export Current Query, Compare Join Strategies

## Ερώτημα #10: Κορυφαία Ζεύγη Ειδών σε Καλλιτέχνες

**Σκοπός:** Εντοπίζει τους 3 πιο συνηθισμένους συνδυασμούς μουσικών ειδών στις ταξινομήσεις καλλιτεχνών.

**Τεχνική Υλοποίηση:** Χρησιμοποιεί Εκφράσεις Κοινού Πίνακα (WITH) για να διαχωρίσει πρώτα συμβολοσειρές ειδών στο διαχωριστικό "/" και στη συνέχεια να αναλύσει τα ζεύγη που προκύπτουν.

**Προσέγγιση Οπτικοποίησης:** Εμφανίζεται ως πινακοειδής αναφορά αναφέροντας με τα δύο είδη που αποτελούν κάθε ζεύγος και πόσοι καλλιτέχνες έχουν το κάθε ζεύγος.

## Αποτελέσματα:

The screenshot shows a database application interface with a sidebar of queries and a main panel for 'Query Results'. The sidebar lists numbered queries from 4 to 13. The main panel displays the results of Query 10: 'Top 3 Genre Pairs in Artists'. The results table has columns: genre1, genre2, and artist\_count. The data shows three pairs: Art-Popular (6 artists), Rock-Pop (4 artists), and Folk-Jazz (2 artists).

genre1	genre2	artist_count
Art	Popular	6
Rock	Pop	4
Folk	Jazz	2

**Query Details:** Lists the top 3 pairs of genres that appear together in artists who performed at festivals.

```

WITH SplitGenres AS (
    SELECT
        p.performance_id,
        a.artist_id,
        a.name AS artist_name,
        CASE
            WHEN a.genre LIKE '%/%' THEN SUBSTRING_INDEX(a.genre, '/', 1)
            ELSE a.genre
        END AS genre1,
        CASE
            WHEN a.genre LIKE '%/%' THEN SUBSTRING_INDEX(a.genre, '/', -1)
            ELSE a.genre
        END AS genre2
    )
    SELECT
        genre1,
        genre2,
        COUNT(*) AS artist_count
    FROM
        SplitGenres
    GROUP BY
        genre1,
        genre2
    ORDER BY
        artist_count DESC
    LIMIT 3;
  
```

Query executed successfully: 3 rows returned. Execution time: 0.0141 seconds

Buttons: Execute Query, Execute With Trace, Compare Query Plans, Export Current Query, Compare Join Strategies

Μια σημαντική πρόκληση που αντιμετωπίσαμε στην υλοποίηση του ερωτήματος #10 ήταν η απαίτηση εύρεσης των κορυφαίων ζευγών μουσικών ειδών. Η αρχική σχεδίαση περιλάμβανε διακριτά πεδία genre και subgenre στον πίνακα Artist. Ωστόσο, κατά την ανάπτυξη διαπιστώσαμε ότι αυτή η αρχιτεκτονική δεν ήταν επαρκής για την αποτύπωση καλλιτεχνών με υβριδικά είδη.

**Πρόβλημα:** Πολλοί καλλιτέχνες δεν εντάσσονται αυστηρά σε ένα κύριο είδος με ένα υποείδος, αλλά συχνά συνδυάζουν δύο ή περισσότερα κύρια είδη (π.χ. "Rock/Folk", "Jazz/Pop").

**Λύση:** Τροποποιήσαμε το σχήμα και την προσέγγιση:

1. Διατηρήσαμε το πεδίο genre αλλά επιτρέψαμε την εισαγωγή συνδυασμών ειδών χωρισμένων με "/"
2. Δημιουργήσαμε το ερώτημα #10 χρησιμοποιώντας Common Table Expressions (WITH) για να:
  - a. Διαχωρίσουμε τις συμβολοσειρές ειδών με διαχωριστικό "/"
  - b. Δημιουργήσουμε όλους τους πιθανούς συνδυασμούς ζευγών
  - c. Συγκεντρώσουμε και μετρήσουμε τις εμφανίσεις κάθε ζεύγους

Αυτή η προσέγγιση επέτρεψε την αποτελεσματική εκτέλεση του ερωτήματος #10 και παρέχει μια πιο ρεαλιστική αναπαράσταση των υβριδικών μουσικών ειδών των σύγχρονων καλλιτεχνών.

### **Ερώτημα #11: Καλλιτέχνες με Λιγότερες Παραστάσεις από τον Κορυφαίο Καλλιτέχνη**

**Σκοπός:** Συγκρίνει τις συχνότητες παραστάσεων καλλιτεχνών για να εντοπίσει αυτούς με τουλάχιστο πέντε φορές λιγότερες συμμετοχές σε φεστιβάλ από τον πιο ενεργό καλλιτέχνη.

**Τεχνική Υλοποίηση:** Χρησιμοποιεί ένα CTE με ένα CROSS JOIN στη μέγιστη τιμή, υπολογίζοντας τη διαφορά για όλους τους άλλους καλλιτέχνες.

**Προσέγγιση Οπτικοποίησης:** Εμφανίζεται ως πινακοειδής αναφορά στην οποία παρουσιάζεται το id, το όνομα και η συμμετοχές σε παραστάσεις του κάθε καλλιτέχνη που έχει λιγότερες παραστάσεις από τον κορυφαίο καλλιτέχνη. Επίσης, παρουσιάζονται οι συμμετοχές του κορυφαίου καλλιτέχνη και η διαφορά ανάμεσα στις παραστάσεις του κάθε καλλιτέχνη με αυτές του κορυφαίου. Ακόμη, εμφανίζονται διάγραμμα πίττας και ραβδόγραμμα τα οποία απεικονίζουν τον αριθμό των συμμετοχών σε φεστιβάλ για κάθε καλλιτέχνη.

## Αποτελέσματα:

Music Festival Database App

Selected Query

- 4. Artist Average Ratings
- 5. Young Artists with Most Festival Participations
- 6. Visitor Attended Performances and Ratings
- 7. Festival Staff with Technical Experience
- 8. Unscheduled Support Staff for a Date
- 9. Visitors with Same Performance Attendance Count
- 10. Top 3 Genre Pairs in Artists
- 11. Artists with Fewer Performances Than Top Artist**
- 12. Staff Required for Each Festival Day
- 13. Artists Who Performed on Multiple Continents

Query Parameters

No parameters required for this query.

Query Details

Show artists who performed at least 5 fewer times than the most active artist.

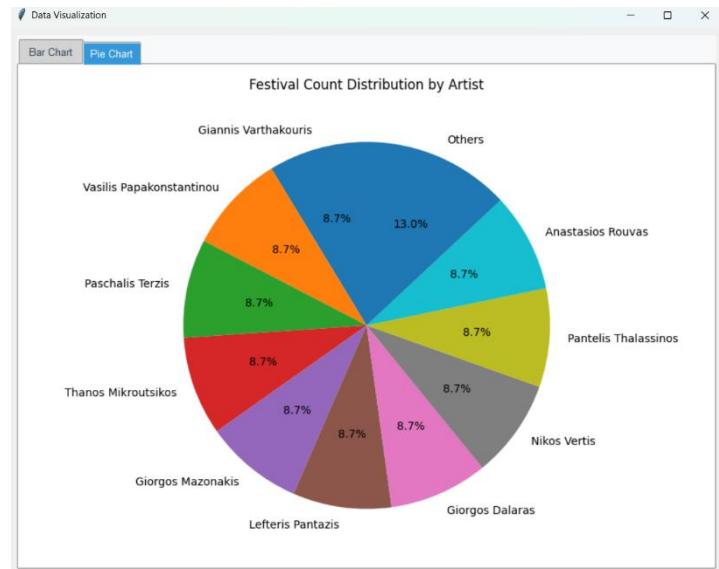
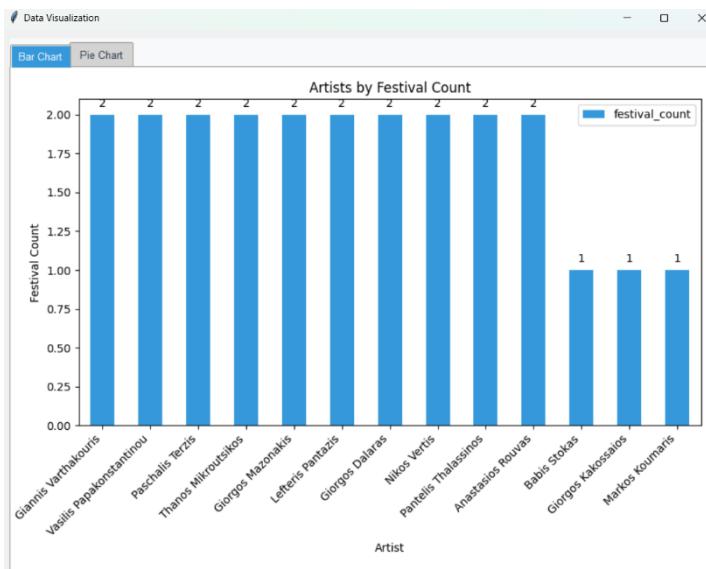
```
WITH ArtistParticipations AS (
    SELECT
        a.artist_id,
        a.name,
        COUNT(DISTINCT f.festival_id) AS festival_count
    FROM
        Artist a
    LEFT JOIN
        Performance p ON a.artist_id = p.artist_id
    LEFT JOIN
        Event e ON p.event_id = e.event_id
    LEFT JOIN
        FestivalDay fd ON e.day_id = fd.day_id
)
```

Query Results

artist_id	name	festival_count	top_artist_count	difference
7	Giorgos Mazonakis	2	7	5
16	Lefteris Pantazis	2	7	5
26	Giorgos Dalaras	2	7	5
5	Nikos Vertis	2	7	5
15	Pantelis Thalassinos	2	7	5
24	Anastasios Rouvas	2	7	5
1	Giannis Varthakouris	2	7	5
9	Vasilis Papakonstantinou	2	7	5
21	Paschalis Terzis	2	7	5
30	Thanos Mikroutsikos	2	7	5
38	Giorgos Kakosalos	1	7	6
36	Markos Kounaris	1	7	6
31	Babis Stokas	1	7	6

Query executed successfully: 13 rows returned. Execution time: 0.0180 seconds

Execute Query | Execute With Trace | Compare Query Plans | Export Current Query | Compare Join Strategies



## Ερώτημα #12: Απαιτούμενο Προσωπικό για Κάθε Ημέρα Φεστιβάλ

**Σκοπός:** Υπολογίζει τις απαιτήσεις στελέχωσης με βάση τις πωλήσεις εισιτηρίων και τις αναλογίες ανά ρόλο.

**Τεχνική Υλοποίηση:** Χρησιμοποιεί μια έκφραση CASE για την εφαρμογή διαφορετικών αναλογιών στελέχωσης με βάση το ρόλο (5% για ασφάλεια, 2% για υποστήριξη).

**Προσέγγιση Οπτικοποίησης:** Εμφανίζεται ως πινακοειδής αναφορά με το όνομα και την ημερομηνία του φεστιβάλ, το προσωπικό που υπάρχει σε κάθε φεστιβάλ και τον ρόλο του

καθενός αλλά και πόσο προσωπικό απαιτείται σε κάθε φεστιβάλ, και τα εισιτήρια που πωλήθηκαν για κάθε φεστιβάλ.

## Αποτελέσματα:

Music Festival Database App																																																																																																																																																																																																																																																																																																								
Select Query																																																																																																																																																																																																																																																																																																								
Query Results																																																																																																																																																																																																																																																																																																								
<a href="#">Export Results</a>				<a href="#">Show Execution Trace</a>				<a href="#">Visualize Data</a>																																																																																																																																																																																																																																																																																																
								210 rows																																																																																																																																																																																																																																																																																																
Query: 12. Staff Required for Each Festival Day   Time: 0.1367s																																																																																																																																																																																																																																																																																																								
<table border="1"> <thead> <tr> <th>festival_id</th><th>festival_name</th><th>staff_n</th><th>assigned_s</th><th>tickets_</th><th>max_staff_per_ven</th><th>avg_staff_per_ven</th><th>operating_ver</th><th>total_concurrent_staff</th><th>staffing_1</th><th>staffing_2</th></tr> </thead> <tbody> <tr><td>2026-09-11</td><td>Rio Summer Sounds</td><td>Securit</td><td>50</td><td>0</td><td>50</td><td>50</td><td>1</td><td>2500</td><td>100</td><td></td></tr> <tr><td>2026-09-11</td><td>Rio Summer Sounds</td><td>Suppor</td><td>20</td><td>0</td><td>20</td><td>20</td><td>1</td><td>400</td><td>100</td><td></td></tr> <tr><td>2026-09-11</td><td>Rio Summer Sounds</td><td>Technic</td><td>3</td><td>0</td><td>1</td><td>1</td><td>1</td><td>3</td><td>300</td><td></td></tr> <tr><td>2026-09-01</td><td>Rio Summer Sounds</td><td>Securit</td><td>50</td><td>0</td><td>50</td><td>50</td><td>2</td><td>5000</td><td>100</td><td></td></tr> <tr><td>2026-09-01</td><td>Rio Summer Sounds</td><td>Suppor</td><td>20</td><td>0</td><td>20</td><td>20</td><td>2</td><td>800</td><td>100</td><td></td></tr> <tr><td>2026-09-01</td><td>Rio Summer Sounds</td><td>Technic</td><td>3</td><td>0</td><td>1</td><td>1</td><td>2</td><td>6</td><td>300</td><td></td></tr> <tr><td>2026-09-01</td><td>Rio Summer Sounds</td><td>Securit</td><td>50</td><td>0</td><td>50</td><td>50</td><td>1</td><td>2500</td><td>100</td><td></td></tr> <tr><td>2026-09-01</td><td>Rio Summer Sounds</td><td>Suppor</td><td>20</td><td>0</td><td>20</td><td>20</td><td>1</td><td>400</td><td>100</td><td></td></tr> <tr><td>2026-09-01</td><td>Rio Summer Sounds</td><td>Technic</td><td>3</td><td>0</td><td>1</td><td>1</td><td>1</td><td>3</td><td>300</td><td></td></tr> <tr><td>2026-09-01</td><td>Rio Summer Sounds</td><td>Securit</td><td>50</td><td>0</td><td>50</td><td>50</td><td>1</td><td>2500</td><td>100</td><td></td></tr> <tr><td>2026-09-01</td><td>Rio Summer Sounds</td><td>Suppor</td><td>20</td><td>0</td><td>20</td><td>20</td><td>1</td><td>400</td><td>100</td><td></td></tr> <tr><td>2026-09-01</td><td>Rio Summer Sounds</td><td>Technic</td><td>3</td><td>0</td><td>1</td><td>1</td><td>1</td><td>3</td><td>300</td><td></td></tr> <tr><td>2026-09-01</td><td>Rio Summer Sounds</td><td>Securit</td><td>4</td><td>5</td><td>4</td><td>4</td><td>1</td><td>80</td><td>100</td><td></td></tr> <tr><td>2026-09-01</td><td>Rio Summer Sounds</td><td>Suppor</td><td>2</td><td>5</td><td>2</td><td>2</td><td>1</td><td>20</td><td>100</td><td></td></tr> <tr><td>2026-09-01</td><td>Rio Summer Sounds</td><td>Technic</td><td>3</td><td>5</td><td>1</td><td>1</td><td>1</td><td>15</td><td>300</td><td></td></tr> <tr><td>2026-09-01</td><td>Rio Summer Sounds</td><td>Securit</td><td>15</td><td>10</td><td>15</td><td>15</td><td>1</td><td>2250</td><td>100</td><td></td></tr> <tr><td>2026-09-01</td><td>Rio Summer Sounds</td><td>Suppor</td><td>6</td><td>10</td><td>6</td><td>6</td><td>1</td><td>360</td><td>100</td><td></td></tr> <tr><td>2026-09-01</td><td>Rio Summer Sounds</td><td>Technic</td><td>3</td><td>10</td><td>1</td><td>1</td><td>1</td><td>30</td><td>300</td><td></td></tr> <tr><td>2025-08-11</td><td>New York World Music Heritage</td><td>Securit</td><td>15</td><td>0</td><td>15</td><td>15</td><td>1</td><td>225</td><td>100</td><td></td></tr> <tr><td>2025-08-11</td><td>New York World Music Heritage</td><td>Suppor</td><td>6</td><td>0</td><td>6</td><td>6</td><td>1</td><td>36</td><td>100</td><td></td></tr> <tr><td>2025-08-11</td><td>New York World Music Heritage</td><td>Technic</td><td>3</td><td>0</td><td>1</td><td>1</td><td>1</td><td>3</td><td>300</td><td></td></tr> <tr><td>2025-08-01</td><td>New York World Music Heritage</td><td>Securit</td><td>50</td><td>0</td><td>50</td><td>50</td><td>1</td><td>2500</td><td>100</td><td></td></tr> <tr><td>2025-08-01</td><td>New York World Music Heritage</td><td>Suppor</td><td>20</td><td>0</td><td>20</td><td>20</td><td>1</td><td>400</td><td>100</td><td></td></tr> <tr><td>2025-08-01</td><td>New York World Music Heritage</td><td>Technic</td><td>3</td><td>0</td><td>1</td><td>1</td><td>1</td><td>3</td><td>300</td><td></td></tr> <tr><td>2025-08-01</td><td>New York World Music Heritage</td><td>Securit</td><td>45</td><td>0</td><td>25</td><td>25</td><td>1</td><td>225</td><td>100</td><td></td></tr> </tbody> </table>											festival_id	festival_name	staff_n	assigned_s	tickets_	max_staff_per_ven	avg_staff_per_ven	operating_ver	total_concurrent_staff	staffing_1	staffing_2	2026-09-11	Rio Summer Sounds	Securit	50	0	50	50	1	2500	100		2026-09-11	Rio Summer Sounds	Suppor	20	0	20	20	1	400	100		2026-09-11	Rio Summer Sounds	Technic	3	0	1	1	1	3	300		2026-09-01	Rio Summer Sounds	Securit	50	0	50	50	2	5000	100		2026-09-01	Rio Summer Sounds	Suppor	20	0	20	20	2	800	100		2026-09-01	Rio Summer Sounds	Technic	3	0	1	1	2	6	300		2026-09-01	Rio Summer Sounds	Securit	50	0	50	50	1	2500	100		2026-09-01	Rio Summer Sounds	Suppor	20	0	20	20	1	400	100		2026-09-01	Rio Summer Sounds	Technic	3	0	1	1	1	3	300		2026-09-01	Rio Summer Sounds	Securit	50	0	50	50	1	2500	100		2026-09-01	Rio Summer Sounds	Suppor	20	0	20	20	1	400	100		2026-09-01	Rio Summer Sounds	Technic	3	0	1	1	1	3	300		2026-09-01	Rio Summer Sounds	Securit	4	5	4	4	1	80	100		2026-09-01	Rio Summer Sounds	Suppor	2	5	2	2	1	20	100		2026-09-01	Rio Summer Sounds	Technic	3	5	1	1	1	15	300		2026-09-01	Rio Summer Sounds	Securit	15	10	15	15	1	2250	100		2026-09-01	Rio Summer Sounds	Suppor	6	10	6	6	1	360	100		2026-09-01	Rio Summer Sounds	Technic	3	10	1	1	1	30	300		2025-08-11	New York World Music Heritage	Securit	15	0	15	15	1	225	100		2025-08-11	New York World Music Heritage	Suppor	6	0	6	6	1	36	100		2025-08-11	New York World Music Heritage	Technic	3	0	1	1	1	3	300		2025-08-01	New York World Music Heritage	Securit	50	0	50	50	1	2500	100		2025-08-01	New York World Music Heritage	Suppor	20	0	20	20	1	400	100		2025-08-01	New York World Music Heritage	Technic	3	0	1	1	1	3	300		2025-08-01	New York World Music Heritage	Securit	45	0	25	25	1	225	100	
festival_id	festival_name	staff_n	assigned_s	tickets_	max_staff_per_ven	avg_staff_per_ven	operating_ver	total_concurrent_staff	staffing_1	staffing_2																																																																																																																																																																																																																																																																																														
2026-09-11	Rio Summer Sounds	Securit	50	0	50	50	1	2500	100																																																																																																																																																																																																																																																																																															
2026-09-11	Rio Summer Sounds	Suppor	20	0	20	20	1	400	100																																																																																																																																																																																																																																																																																															
2026-09-11	Rio Summer Sounds	Technic	3	0	1	1	1	3	300																																																																																																																																																																																																																																																																																															
2026-09-01	Rio Summer Sounds	Securit	50	0	50	50	2	5000	100																																																																																																																																																																																																																																																																																															
2026-09-01	Rio Summer Sounds	Suppor	20	0	20	20	2	800	100																																																																																																																																																																																																																																																																																															
2026-09-01	Rio Summer Sounds	Technic	3	0	1	1	2	6	300																																																																																																																																																																																																																																																																																															
2026-09-01	Rio Summer Sounds	Securit	50	0	50	50	1	2500	100																																																																																																																																																																																																																																																																																															
2026-09-01	Rio Summer Sounds	Suppor	20	0	20	20	1	400	100																																																																																																																																																																																																																																																																																															
2026-09-01	Rio Summer Sounds	Technic	3	0	1	1	1	3	300																																																																																																																																																																																																																																																																																															
2026-09-01	Rio Summer Sounds	Securit	50	0	50	50	1	2500	100																																																																																																																																																																																																																																																																																															
2026-09-01	Rio Summer Sounds	Suppor	20	0	20	20	1	400	100																																																																																																																																																																																																																																																																																															
2026-09-01	Rio Summer Sounds	Technic	3	0	1	1	1	3	300																																																																																																																																																																																																																																																																																															
2026-09-01	Rio Summer Sounds	Securit	4	5	4	4	1	80	100																																																																																																																																																																																																																																																																																															
2026-09-01	Rio Summer Sounds	Suppor	2	5	2	2	1	20	100																																																																																																																																																																																																																																																																																															
2026-09-01	Rio Summer Sounds	Technic	3	5	1	1	1	15	300																																																																																																																																																																																																																																																																																															
2026-09-01	Rio Summer Sounds	Securit	15	10	15	15	1	2250	100																																																																																																																																																																																																																																																																																															
2026-09-01	Rio Summer Sounds	Suppor	6	10	6	6	1	360	100																																																																																																																																																																																																																																																																																															
2026-09-01	Rio Summer Sounds	Technic	3	10	1	1	1	30	300																																																																																																																																																																																																																																																																																															
2025-08-11	New York World Music Heritage	Securit	15	0	15	15	1	225	100																																																																																																																																																																																																																																																																																															
2025-08-11	New York World Music Heritage	Suppor	6	0	6	6	1	36	100																																																																																																																																																																																																																																																																																															
2025-08-11	New York World Music Heritage	Technic	3	0	1	1	1	3	300																																																																																																																																																																																																																																																																																															
2025-08-01	New York World Music Heritage	Securit	50	0	50	50	1	2500	100																																																																																																																																																																																																																																																																																															
2025-08-01	New York World Music Heritage	Suppor	20	0	20	20	1	400	100																																																																																																																																																																																																																																																																																															
2025-08-01	New York World Music Heritage	Technic	3	0	1	1	1	3	300																																																																																																																																																																																																																																																																																															
2025-08-01	New York World Music Heritage	Securit	45	0	25	25	1	225	100																																																																																																																																																																																																																																																																																															
Query executed successfully: 210 rows returned. Execution time: 0.1367 seconds																																																																																																																																																																																																																																																																																																								
<a href="#">Execute Query</a>	<a href="#">Execute With Trace</a>	<a href="#">Compare Query Plans</a>	<a href="#">Export Current Query</a>	<a href="#">Compare Join Strategies</a>																																																																																																																																																																																																																																																																																																				

**Ερώτημα #13:** Καλλιτέχνες που Εμφανίστηκαν σε Πολλαπλές Ηπείρους

**Σκοπός:** Εντοπίζει διεθνώς επιτυχημένους καλλιτέχνες που έχουν εμφανιστεί σε τουλάχιστον τρεις διαφορετικές ηπείρους.

**Τεχνική Υλοποίηση:** Χρησιμοποιεί GROUP\_CONCAT για την αποτελεσματική συλλογή λιστών ηπείρων εντός του ερωτήματος.

**Προσέγγιση Οπτικοποίησης:** Εμφανίζεται ως πινακοειδής αναφορά με το id και το όνομα του καλλιτέχνη, σε πόσες και σε ποιες ηπείρους εμφανίστηκε. Επίσης, εμφανίζονται γραφήματα που απεικονίζουν σε πόσες ηπείρους εμφανίστηκε ο κάθε καλλιτέχνης.

## Αποτελέσματα:

Music Festival Database App

Select Query

```
4 Artist Average Ratings
5 Young Artists with Most Festival Participations
6 Visitor Attended Performances and Ratings
7 Festival with Lowest Technical Staff Experience
8 Unscheduled Support Staff for a Date
9 Visitors with Same Performance Attendance Count
10 Top 3 Genre Pairs in Artists
11 Artists with Fewer Performances Than Top Artist
12 Staff Required for Each Festival Day
13 Artists Who Performed on Multiple Continents
```

Query Parameters

No parameters required for this query.

Query Details

Finds artists who performed in festivals on at least 3 different continents.

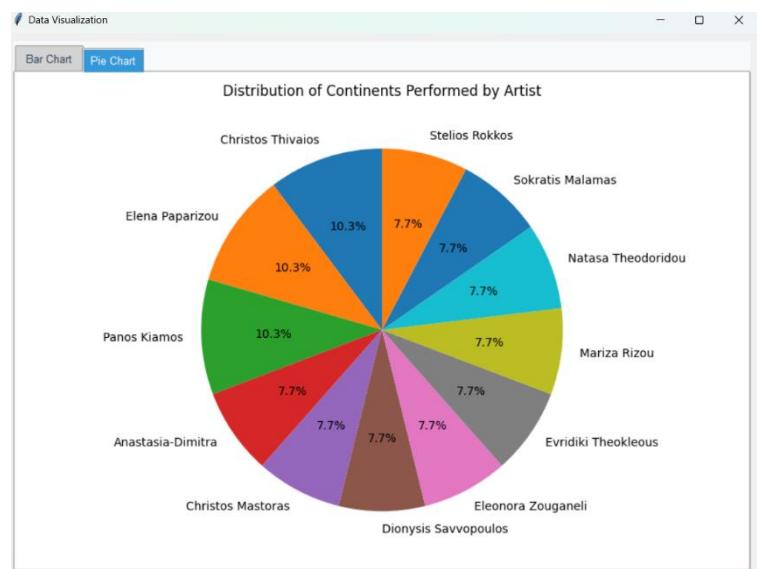
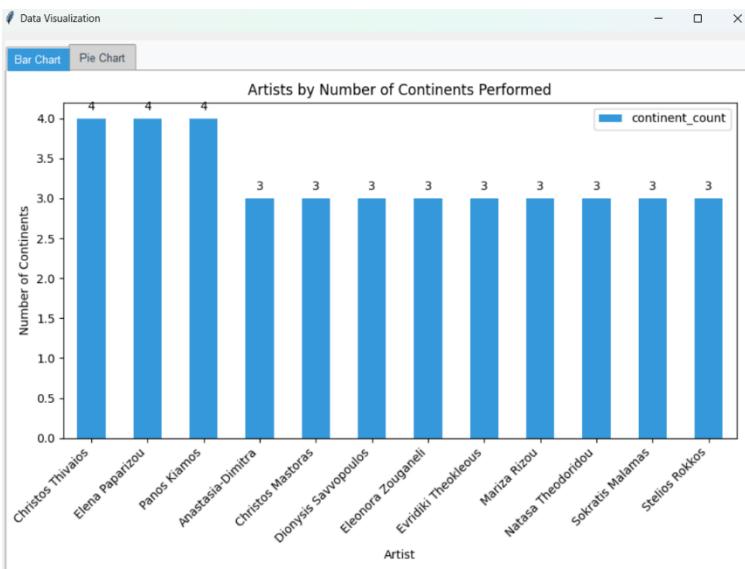
```
SELECT
    a.artist_id,
    a.name,
    COUNT(DISTINCT l.continent) AS continent_count,
    GROUP_CONCAT(DISTINCT l.continent) AS continents
FROM
    Artist a
JOIN
    Performance p ON a.artist_id = p.artist_id
JOIN
    Event e ON p.event_id = e.event_id
JOIN
    FestivalDay fd ON e.day_id = fd.day_id
```

Query Results

artist_id	name	continent_count	continents
33	Christos Thivaios	4	Australia,Europe,North America,South America
25	Elena Paparizou	4	Australia,Europe,North America,South America
17	Panos Kiamos	4	Australia,Europe,North America,South America
39	Anastasia-Dimitra	3	Europe,North America,South America
34	Christos Mastoras	3	Australia,Europe,South America
10	Dionysis Savvopoulos	3	Europe,North America,South America
6	Eleonora Zouganelli	3	Europe,North America,South America
13	Evdiki Theokleous	3	Europe,North America,South America
28	Mariza Rizou	3	Europe,North America,South America
18	Natasa Theodoridou	3	Europe,North America,South America
32	Sokratis Malamas	3	Australia,Europe,South America
20	Stelios Rokkos	3	Europe,North America,South America

Query executed successfully: 12 rows returned Execution time: 0.0194 seconds

Execute Query | Execute With Trace | Compare Query Plans | Export Current Query | Compare Join Strategies



## Ερώτημα #14: Είδη με Συνεπείς Αριθμούς Παραστάσεων

**Σκοπός:** Εντοπίζει μουσικά είδη που διατηρούν σταθερή δημοτικότητα σε διαδοχικά έτη, με τουλάχιστον 3 εμφανίσεις ανά έτος.

**Τεχνική Υλοποίηση:** Χρησιμοποιεί ένα CTE για τον υπολογισμό ετήσιων αριθμών παραστάσεων και στη συνέχεια αυτοσυνδέεται για την εύρεση αντίστοιχων αριθμών σε διαδοχικά έτη.

**Προσέγγιση Οπτικοποίησης:** Εμφανίζεται ως πινακοειδής αναφορά με τα μουσικά είδη, τα έτη που εμφανίζουν συνέπεια και τον αριθμό των παραστάσεων που εμφανίστηκαν.

### Αποτελέσματα:

The screenshot shows a 'Music Festival Database App' interface. On the left, a 'Select Query' sidebar lists numbered queries from 6 to 15. Query 14 is selected, titled 'Genres with Consistent Performance Counts'. The main area is titled 'Query Results' and displays a table with one row. The table has columns: genre, year1, year2, and performance\_count. The data is: Pop/Electronic, 2024, 2025, 4. Below the table, the SQL query is shown:

```
WITH GenreYearCounts AS (
    SELECT
        a.genre,
        f.year,
        COUNT(DISTINCT p.performance_id) AS performance_count
    FROM
        Artist a
    JOIN
        Performance p ON a.artist_id = p.artist_id
    JOIN
        Event e ON p.event_id = e.event_id
    JOIN
        FestivalDay fd ON e.day_id = fd.day_id
)
```

At the bottom, a message says 'Query executed successfully. 1 rows returned. Execution time: 0.0111 seconds' and there are buttons for 'Execute Query', 'Execute With Trace', etc.

### Ερώτημα #15: Κορυφαίοι Επισκέπτες με Βάση τις Βαθμολογίες για έναν Καλλιτέχνη

**Σκοπός:** Εντοπίζει τους πέντε επισκέπτες που έδωσαν τις υψηλότερες βαθμολογίες σε έναν συγκεκριμένο καλλιτέχνη.

**Τεχνική Υλοποίηση:** Υπολογίζει μια συνολική βαθμολογία από μεμονωμένες διαστάσεις βαθμολόγησης και κατατάσσει τους επισκέπτες με βάση αυτό το συνδυασμένο μετρικό.

### Προσέγγιση Οπτικοποίησης:

Εμφανίζεται ως πινακοειδής αναφορά με τα id και τα ονόματα των επισκεπτών που άφησαν κριτικές για τον συγκεκριμένο καλλιτέχνη, πόσες κριτικές άφησαν, τον συνολικό βαθμό και τον μέσο όρο της αξιολόγησης που έκανε ο κάθε επισκέπτης.

## Αποτελέσματα:

The screenshot shows a database application window titled "Music Festival Database App". On the left, there's a "Select Query" pane with a list of numbered queries. Item 15, "Top Visitors by Ratings for an Artist", is selected and highlighted in blue. Below it, "Query Parameters" show "artist\_id: 28". The main area is titled "Query Results" and displays a table with 5 rows. The columns are "visitor\_id", "visitor\_name", "artist\_name", "total\_score", "review\_count", and "avg\_overall\_rating". The data is as follows:

visitor_id	visitor_name	artist_name	total_score	review_count	avg_overall_rating
53	Eva Nikolau	Mariza Rizou	25	1	5.0000
74	Stelios Petridis	Mariza Rizou	25	1	5.0000
56	Anna Sawidis	Mariza Rizou	25	1	5.0000
51	Panagiotis Michailidis	Mariza Rizou	25	1	5.0000
72	Alexandros Galanis	Mariza Rizou	25	1	5.0000

Below the table, the "Query Details" pane contains the SQL query:

```
SELECT
    v.visitor_id,
    CONCAT(v.first_name, ' ', v.last_name) AS visitor_name,
    a.artist_name,
    SUM(r.artistic_rating + r.sound_rating + r.stage_rating +
    r.organization_rating + r.overall_rating) AS total_score,
    COUNT(r.review_id) AS review_count,
    AVG(r.overall_rating) AS avg_overall_rating
FROM
    Visitor v
JOIN
    Review r ON v.visitor_id = r.visitor_id
JOIN
    Artist a ON v.artist_id = a.artist_id
ORDER BY total_score DESC
LIMIT 5;
```

The status bar at the bottom indicates: "Query executed successfully: 5 rows returned. Execution time: 0.0115 seconds".

## 7. Σύνοψη Τεχνικών Βελτιστοποίησης

Από την ανάπτυξη και βελτιστοποίηση της βάσης δεδομένων Pulse University, αποκτήσαμε πολύτιμες γνώσεις σχετικά με τις τεχνικές βελτιστοποίησης ερωτημάτων:

### Η Αξία της Ανάλυσης Ιχνους Εκτέλεσης

Η μελέτη των ιχνών εκτέλεσης μας επέτρεψε να:

- Κατανοήσουμε τις αποφάσεις του βελτιστοποιητή ερωτημάτων
- Εντοπίσουμε περιπτώσεις όπου αγνοούνται χρήσιμα ευρετήρια
- Προσδιορίσουμε τους παράγοντες που επηρεάζουν την απόδοση

Για παράδειγμα, στο ερώτημα #4, ανακαλύψαμε ότι ο βελτιστοποιητής επέλεγε λανθασμένα να μην χρησιμοποιήσει το ευρετήριο idx\_review\_performance, οδηγώντας σε πλήρη σάρωση του πίνακα Review.

### Στρατηγικές Βελτιστοποίησης με Εφαρμογή στην Πράξη

Από τα πειράματά μας, καταλήξαμε στις ακόλουθες βέλτιστες πρακτικές:

1. **Επιλεκτική χρήση FORCE INDEX:** Είναι χρήσιμη όταν ο βελτιστοποιητής επιλέγει λανθασμένα ευρετήρια, αλλά πρέπει να εφαρμόζεται με προσοχή.

2. **Συνδυασμός Hash Join με ευρετήρια:** Η στρατηγική Hash Join με κατάλληλα ευρετήρια παρέχει τη βέλτιστη απόδοση για πολύπλοκες συνδέσεις πινάκων.
3. **Ευρετήρια επιλεγμένα βάσει προτύπων χρήσης:** Τα ευρετήρια πρέπει να σχεδιάζονται με βάση τα συγκεκριμένα ερωτήματα που θα εκτελούνται συχνά.
4. **Balance μεταξύ ευρετηρίων και χώρου αποθήκευσης:** Η προσθήκη περισσότερων ευρετηρίων βελτιώνει την απόδοση ερωτημάτων αλλά αυξάνει τις απαιτήσεις αποθήκευσης και το κόστος ενημέρωσης.

## 8. Περίληψη Πλαισίου Δοκιμών

### Μεθοδολογία Δοκιμών:

Για την εξασφάλιση της αξιοπιστίας και της ορθότητας του συστήματός μας, αναπτύξαμε ένα ολοκληρωμένο αυτοματοποιημένο πλαίσιο δοκιμών που αξιολογεί συστηματικά όλες τις πτυχές της ακεραιότητας της βάσης δεδομένων. Ο κώδικας για τις δοκιμές βρίσκεται στο αρχείο test.sql και τα αποτελέσματα στο test\_results.md. Η προσέγγισή μας περιλαμβάνει:

- **Απομονωμένες Περιπτώσεις Δοκιμών:** Κάθε δοκιμή εκτελείται εντός της δικής της συναλλαγής για την αποφυγή παρεμβολών
- **Αυτόματο Καθαρισμό:** Όλα τα δεδομένα δοκιμών διαγράφονται αυτόματα μετά την εκτέλεση
- **Παρακολούθηση Απόδοσης:** Οι δοκιμές μετρούν τον χρόνο εκτέλεσης για τον εντοπισμό πιθανών προβλημάτων
- **Λεπτομερή Καταγραφή:** Καταγράφεται η κατάσταση επιτυχίας/αποτυχίας και τα μηνύματα σφάλματος για κάθε δοκιμή

### Κατηγορίες περιορισμών που δοκιμάστηκαν:

Η εκτέλεση των δοκιμών μας επικυρώνει τους ακόλουθους τύπους περιορισμών:

1. Περιορισμοί Πρωτεύοντος Κλειδιού και Μοναδικότητας
2. Περιορισμοί Ξένου Κλειδιού
3. Περιορισμοί Ελέγχου (έλεγχοι πεδίου)
4. Σύνθετοι Επιχειρηματικοί Κανόνες μέσω Σκανδαλών (Triggers)
5. Περιορισμοί Απαιτήσεων Προσωπικού
6. Δοκιμές Αντοχής Συστήματος

### Βασικά Ευρήματα:

Η διαδικασία δοκιμών έδωσε πολύτιμες γνώσεις για την υλοποίηση της βάσης δεδομένων μας:

1. **Αλληλεξαρτήσεις Σκανδαλών:** Οι σύνθετοι περιορισμοί αλληλοεπιδρούν μερικές φορές με μη προφανείς τρόπους
2. **Εξισορρόπηση Περιορισμών και Απόδοσης:** Η εφαρμογή σύνθετων επιχειρηματικών κανόνων απαιτεί ισορροπία
3. **Σημασία Οριακών Περιπτώσεων:** Πολλά πιθανά ζητήματα ακεραιότητας δεδομένων εντοπίστηκαν μέσω δοκιμών οριακών περιπτώσεων
4. **Απομόνωση Συναλλαγών:** Διασφαλίζοντας την απομόνωση των δοκιμών αναδείχθηκε η σημασία της σωστής διαχείρισης συναλλαγών

Όλες οι δοκιμές ολοκληρώθηκαν με επιτυχία, επιδεικνύοντας την ακεραιότητα της υλοποίησής μας.

## 9. Εντολές εύρεσης δεδομένων για εισαγωγή στα ερωτήματα από τον χρήστη

Εύρεση καλλιτεχνών που έχουν αξιολογήσεις:

```
SELECT DISTINCT
    p.artist_id,
    a.name AS artist_name,
    COUNT(r.review_id) AS review_count,
    AVG(r.artist_rating) AS avg_artist_rating,
    AVG(r.overall_rating) AS avg_overall_rating
FROM
    Performance p
JOIN
    Review r ON p.performance_id = r.performance_id
JOIN
    Artist a ON p.artist_id = a.artist_id
WHERE
    p.artist_id IS NOT NULL
GROUP BY
    p.artist_id, a.name
ORDER BY
    review_count DESC, avg_overall_rating DESC;
```

Εύρεση επισκεπτών που έκαναν αξιολογήσεις:

```
SELECT DISTINCT visitor_id
FROM Review;
```

Εύρεση 10 ημερομηνιών στις οποίες υπάρχει προσωπικό υποστήριξης χωρίς προγραμματισμένη εργασία:

```
SELECT
    fd.festival_date,
    f.name AS festival_name,
    f.year AS festival_year,
    COUNT(DISTINCT s.staff_id) AS available_support_staff
FROM
    FestivalDay fd
JOIN
    Festival f ON fd.festival_id = f.festival_id
JOIN
    Staff s ON s.role_id = 3 -- Support staff
LEFT JOIN (
    -- Get all staff assignments on each date
    SELECT
        sa.staff_id,
        fd2.festival_date
    FROM
        Staff_Assignment sa
    JOIN
        Event e ON sa.event_id = e.event_id
    JOIN
        FestivalDay fd2 ON e.day_id = fd2.day_id
)
```

```
JOIN
Staff s2 ON sa.staff_id = s2.staff_id
WHERE
s2.role_id = 3 -- Support staff
) AS assigned_staff ON s.staff_id = assigned_staff.staff_id AND fd.festival_date =
assigned_staff.festival_date
WHERE
assigned_staff.staff_id IS NULL -- Only include staff who have no assignments on this
date
GROUP BY
fd.festival_date, f.name, f.year
ORDER BY
available_support_staff DESC, -- Dates with most available staff first
fd.festival_date ASC
LIMIT 10;
```