Βάσεις Δεδομένων

Βασίλης Αφεντουλίδης - el22181
Θανάσης Γιαννόπουλος - el22054
Γιάννης Λεβέντης - el18168

Μάιος 2025

Εισαγωγή

Η παρούσα εργασία αφορά τον σχεδιασμό και την υλοποίηση ενός ολοκληρωμένου συστήματος διαχείρισης βάσης δεδομένων για το διεθνές φεστιβάλ μουσικής του Πανεπιστημίου Pulse. Το σύστημα υποστηρίζει τη διαχείριση φεστιβάλ, εκδηλώσεων, καλλιτεχνών, εισιτηρίων, αξιολογήσεων και προσωπικού, καθώς και την εφαρμογή πολύπλοκων επιχειρηματικών κανόνων.

Constraints

Το σύστημα διαχείρισης του φεστιβάλ Pulse University βασίζεται στις εξής απαιτήσεις:

- **Φεστιβάλ:** Διεξαγωγή ετησίως, σε μία ή περισσότερες συνεχόμενες ημέρες, σε διαφορετική τοποθεσία κάθε έτος
- **Σκηνές:** Κάθε σκηνή διαθέτει όνομα, περιγραφή, χωρητικότητα και απαιτούμενο τεχνικό εξοπλισμό
- **Παραστάσεις:** Πραγματοποίηση σε συγκεκριμένες σκηνές με μέγιστη διάρκεια 3 ωρών και υποχρεωτικά διαλείμματα 5-30 λεπτών
- **Καλλιτέχνες:** Διαχείριση σόλο καλλιτεχνών και συγκροτημάτων, με περιορισμούς στις εμφανίσεις (όχι ταυτόχρονες εμφανίσεις, μέγιστο 3 συνεχή έτη)
- **Εισιτήρια:** Ηλεκτρονική πώληση με περιορισμούς χωρητικότητας και δυνατότητα μεταπώλησης
- **Προσωπικό:** Τεχνικό, ασφαλείας και βοηθητικό, με συγκεκριμένες αναλογίες σε σχέση με τους επισκέπτες (5
- **Αξιολογήσεις:** Δυνατότητα αξιολόγησης παραστάσεων από τους επισκέπτες με κλίμακα Likert σε διάφορες κατηγορίες

Σχεδιασμός Βάσης Δεδομένων

Περιορισμοί και Indexes

Περιορισμοί Ακεραιότητας

- **Περιορισμοί κλειδιών:** Πρωτεύοντα κλειδιά με AUTO_INCREMENT
- **Αναφορική ακεραιότητα:** Foreign keys με κατάλληλες ενέργειες CASCADE/RESTRICT
- **Περιορισμοί πεδίου τιμών:**
  - Διάρκεια παράστασης 180 λεπτά
  - Διάρκεια διαλείμματος: 5-30 λεπτά
  - Αξιολογήσεις με κλίμακα 1-5
- **Μοναδικοί περιορισμοί:**
  - Μοναδικό έτος φεστιβάλ
  - Ένα εισιτήριο ανά επισκέπτη για κάθε παράσταση
  - Μοναδικό ζεύγος (σκηνή, ώρα έναρξης)

Indexes

- **Πρωτεύοντα ευρετήρια:** Για όλα τα πρωτεύοντα κλειδιά
- **Ευρετήρια ξένων κλειδιών:** π.χ. fk_Performance_Artist, idx_performance_event
- **Σύνθετα ευρετήρια:**
  - idx_ticket_visitor (Visitor_ID, Event_ID)
  - idx_rating_performance (Performance_ID, Visitor_ID)
  - uq_stage_starttime (Stage_ID, Start_Time)

Business Rules

Οι επιχειρηματικοί κανόνες υλοποιήθηκαν μέσω συνδυασμού περιορισμών, triggers και stored procedures:

Διαχείριση Φεστιβάλ και Παραστάσεων

- Επιτρέπεται μόνο ένα φεστιβάλ ανά έτος (UNIQUE INDEX στο Start_Year)
- Έλεγχος επικαλυπτόμενων χρόνων εμφανίσεων (trigger trg_check_performance_overlap)
- Έλεγχος διάρκειας διαλείμματος (trigger trg_check_break_duration)
- Έλεγχος συμμετοχής καλλιτεχνών/συγκροτημάτων (trigger trg_check_artist_availability)
- Περιορισμός συμμετοχών σε 3 συνεχή έτη (stored procedure sp_check_artist_continuity)

Διαχείριση Εισιτηρίων και Επισκεπτών

- Έλεγχος χωρητικότητας σκηνής (trigger trg_check_stage_capacity)
- Περιορισμός VIP εισιτηρίων στο 10% (trigger trg_check_vip_ratio)
- Έλεγχος ενεργοποίησης εισιτηρίων (trigger trg_activate_ticket)
- Υποστήριξη μεταπώλησης εισιτηρίων (stored procedures sp_offer_ticket_resale, sp_express_buyer_interest, sp_automatch_resale)

Διαχείριση Προσωπικού

- Έλεγχος αναλογίας προσωπικού ασφαλείας (5%) και βοηθητικού προσωπικού (2%) σε σχέση με τους επισκέπτες (trigger trg_check_personnel_ratio)

Views

Για την απλοποίηση πρόσβασης στα δεδομένα και την κάλυψη συχνών αναγκών, υλοποιήθηκαν οι εξής όψεις (views):

Στατιστικά και Αναλύσεις

- **festival_performance_statistics:** Συγκεντρωτικά στοιχεία εμφανίσεων και αξιολογήσεων ανά φεστιβάλ
- **artist_festival_history:** Ιστορικό συμμετοχών και αξιολογήσεων καλλιτεχνών
- **visitor_ratings_summary:** Συνοπτική παρουσίαση αξιολογήσεων ανά επισκέπτη

Λειτουργικές Όψεις

- **upcoming_performances:** Προβολή επερχόμενων παραστάσεων
- **safety_compliance_status:** Έλεγχος συμμόρφωσης με κανόνες ασφαλείας
- **ticket_availability:** Διαθεσιμότητα εισιτηρίων ανά εκδήλωση

Η χρήση των views προσφέρει σημαντικά πλεονεκτήματα:

- Απλοποίηση πολύπλοκων ερωτημάτων
- Ενιαία πρόσβαση σε δεδομένα που απαιτούν συνενώσεις πολλαπλών πινάκων
- Ενίσχυση ασφάλειας μέσω περιορισμένης πρόσβασης στα δεδομένα

Web App

Η διαδικτυακή εφαρμογή υλοποιήθηκε με τεχνολογίες Node.js, Express και EJS για την παροχή διεπαφής στους χρήστες του συστήματος.

Αρχιτεκτονική Εφαρμογής

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

- **Models:** Αλληλεπίδραση με τη βάση δεδομένων
- **Views:** Πρότυπα EJS για την παρουσίαση των δεδομένων
- **Controllers:** Χειρισμός αιτημάτων και επιχειρηματική λογική
- **Routes:** Δρομολόγηση αιτημάτων στους κατάλληλους controllers

Λειτουργίες Εφαρμογής

- Προβολή φεστιβάλ, εκδηλώσεων και παραστάσεων
- Αγορά και μεταπώληση εισιτηρίων
- Καταχώριση και προβολή αξιολογήσεων
- Διαχείριση προσωπικού και εξοπλισμού
- Εκτέλεση ερωτημάτων και προβολή στατιστικών

Ερωτήματα Q04 και Q06

Για κάθε ερώτημα εφαρμόστηκαν διαφορετικές στρατηγικές βελτιστοποίησης και αλγόριθμοι συνένωσης (join algorithms) με στόχο τη σύγκριση της αποδοτικότητάς τους.

Ερώτημα Q04: Μέσος Όρος Αξιολογήσεων Καλλιτέχνη

Ανάλυση επίδοσης

Στρατηγική	Χρόνος (sec)	Βελτίωση	Σχόλια
Αρχικό Ερώτημα	0,00613	-	Απόφαση βελτιστοποιητή
Force Index	0,00168	72,6%	Αξιοποίηση ευρετηρίου fk_Performance_Artist
Straight Join	0,00132	78,5%	Καθορισμένη σειρά πινάκων
Merge Join	0,00097	84,1%	Βέλτιστη επίδοση
Hash Join	0,00115	81,2%	Αποδοτικό για μικρούς πίνακες

Πίνακας 1: Συγκριτική ανάλυση χρόνων εκτέλεσης για το ερώτημα Q04

Ανάλυση πλάνου εκτέλεσης

Η ανάλυση των traces αποκαλύπτει ότι όλες οι στρατηγικές ακολουθούν παρόμοια μονοπάτια πρόσβασης:

- **Πίνακας Artist (a):**
  - access_type: "const" (βέλτιστος τρόπος πρόσβασης)
  - key: "PRIMARY"
  - rows: 1 (άμεση πρόσβαση μέσω πρωτεύοντος κλειδιού)
  - filtered: 100% (δεν απαιτεί επιπλέον φιλτράρισμα)
- **Πίνακας Performance (p):**
  - access_type: "ref"
  - key: "fk_Performance_Artist"
  - rows: 1 (εκτιμώμενος αριθμός γραμμών)
  - χρήση του ευρετηρίου του ξένου κλειδιού
  - using_index: true (covered index - βελτιστοποιημένη πρόσβαση)
- **Πίνακας Rating (r):**
  - access_type: "ref"
  - key: "idx_rating_performance"
  - rows: 1 (εκτιμώμενος αριθμός γραμμών ανά παράσταση)

Είναι αξιοσημείωτο ότι παρά το σχεδόν πανομοιότυπο πλάνο εκτέλεσης, οι χρόνοι εκτέλεσης διαφέρουν σημαντικά. Αυτό οφείλεται κυρίως στον τρόπο με τον οποίο εκτελείται η συνένωση σε χαμηλότερο επίπεδο.

Συμπεράσματα Q04

1. Η στρατηγική **Merge Join** προσφέρει την καλύτερη επίδοση για αυτό το ερώτημα, με βελτίωση 84,1% σε σχέση με το αρχικό ερώτημα.

2. Παρά το παρόμοιο πλάνο εκτέλεσης, οι διαφορετικές υποδείξεις επηρεάζουν σημαντικά την απόδοση, πιθανώς λόγω διαφορών στον τρόπο με τον οποίο αξιοποιούνται τα ευρετήρια και τη διαχείριση της μνήμης.

3. Η επιλογή των κατάλληλων ευρετηρίων (idx_rating_performance, fk_Performance_Artist) συμβάλλει σημαντικά στη βελτιστοποίηση του ερωτήματος.

4. Το μικρό μέγεθος πινάκων και η υψηλή επιλεκτικότητα των συνθηκών (ακριβώς 1 καλλιτέχνης) καθιστούν αποδοτικές και τις τρεις στρατηγικές συνένωσης.

Ερώτημα Q06: Ιστορικό Παραστάσεων και Αξιολογήσεις Επισκέπτη

Ανάλυση επίδοσης

Το ερώτημα Q06 είναι σημαντικά πιο περίπλοκο από το Q04, καθώς περιλαμβάνει συνενώσεις 8 πινάκων (Visitor, Ticket, Event, Performance, Stage, Performance_Type, Artist/Band, Rating) και ταξινόμηση (filesort) των αποτελεσμάτων.

Στρατηγική	Χρόνος (sec)	Βελτίωση	Σχόλια
Αρχικό Ερώτημα	0,00842	-	Πολύπλοκο πλάνο με nested loops
Force Index	0,00631	25,1%	Επιβολή idx_ticket_visitor, idx_performance_event
Straight Join	0,00748	11,2%	Μικρή βελτίωση
Merge Join	0,00975	-15,8%	Χειρότερη επίδοση, πολλοί πίνακες
Hash Join	0,01246	-48,0%	Υψηλό κόστος κατασκευής hash table

Πίνακας 2: Συγκριτική ανάλυση χρόνων εκτέλεσης για το ερώτημα Q06

Ανάλυση πλάνου εκτέλεσης

Το πλάνο εκτέλεσης του Q06 περιλαμβάνει πολύπλοκες συνενώσεις και προσωρινούς πίνακες:

- **Προσωρινός πίνακας με filesort:** Ταξινόμηση κατά p.Start_Time desc
- **Αλληλουχία προσπελάσεων:**
  - Visitor (v): access_type "const" μέσω PRIMARY key
  - Ticket (t): access_type "ref" μέσω uq_ticket_visitor_performance ή idx_ticket_visitor
  - Event (e): access_type "eq_ref" μέσω PRIMARY key
  - Performance (p): access_type "ref" μέσω uq_stage_starttime ή idx_performance_event
  - Stage (s): access_type "eq_ref" μέσω PRIMARY key
- **Conditional access:**
  - Artist (a): eq_ref με συνθήκη p.Artist_ID is not null

  - Band (b): eq_ref με συνθήκη p.Band_ID is not null

- **Block nested-loop join για Performance_Type:**
  - access_type "ALL" (πλήρης σάρωση πίνακα)
  - buffer_size "12kb" (μικρός buffer λόγω μικρού μεγέθους πίνακα)
- **Υποερώτημα:** Για την εύρεση του Status_ID που αντιστοιχεί στο 'activated'

Διαφορές μεταξύ στρατηγικών

- **Force Index:** Αναγκάζει τη χρήση των ευρετηρίων idx_ticket_visitor και idx_performance_event αντί των uq_ticket_visitor_performance και uq_stage_starttime. Αυτή η αλλαγή βελτιώνει την απόδοση κατά 25%.
- **Straight Join:** Επιβάλλει τη σειρά συνένωσης των πινάκων, με αποτέλεσμα μικρότερη βελτίωση σε σύγκριση με το Force Index.
- **Merge Join:** Απαιτεί ταξινομημένα δεδομένα, κάτι που αυξάνει το κόστος λόγω των πολλαπλών πινάκων που συμμετέχουν στο ερώτημα.
- **Hash Join:** Η κατασκευή των hash tables για πολλούς πίνακες οδηγεί σε χειρότερη επίδοση για αυτό το πολύπλοκο ερώτημα.

Συμπεράσματα Q06

1. Η στρατηγική **Force Index** αποδίδει καλύτερα για το πολύπλοκο ερώτημα Q06, με συγκεκριμένα επιλεγμένα ευρετήρια.
2. Σε αντίθεση με το Q04, οι στρατηγικές Merge Join και Hash Join επιδεινώνουν την απόδοση του Q06, καθώς το κόστος δημιουργίας των δομών δεδομένων που απαιτούνται υπερβαίνει το όφελος από τη βελτιστοποίηση των συνενώσεων.
3. Ο αριθμός των πινάκων που συμμετέχουν στο ερώτημα (8) επηρεάζει σημαντικά την επίδοση των διαφορετικών στρατηγικών.
4. Η επιλογή του κατάλληλου ευρετηρίου (idx_ticket_visitor έναντι uq_ticket_visitor_performance) έχει σημαντικό αντίκτυπο στην απόδοση.
5. Η σειρά συνένωσης των πινάκων είναι κρίσιμη, με βέλτιστη σειρά αυτή που ξεκινά από τον επισκέπτη (υψηλή επιλεκτικότητα) και συνεχίζει στους σχετικούς πίνακες.

Συγκριτική Ανάλυση και Συμπεράσματα

- **Nested Loop Join:** Αποδοτική για μικρούς πίνακες και υψηλή επιλεκτικότητα, όπως στο Q04 και Q06.
- **Merge Join:** Εξαιρετική απόδοση για το απλούστερο Q04, αλλά χειρότερη για το πολύπλοκο Q06 λόγω του κόστους ταξινόμησης πολλαπλών πινάκων.
- **Hash Join:** Καλή απόδοση για το Q04 με λίγους πίνακες, αλλά σημαντική μείωση απόδοσης για το Q06 με πολλούς πίνακες.