ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ

ΒΑΣΕΙΣ ΔΕΔΟΜΕΝΩΝ

2025

Η παρούσα αναφορά πραγματεύεται την εργασία στα πλαίσια του μαθήματος ΒΑΣΕΙΣ ΔΕΔΟΜΕΝΩΝ στο έτος 2024-2025 από τους φοιτητές Παντελεήμων Αγγελίδη, Ηλία Πασχόπουλου και Ιάσωνα Περαντζάκη.

+ Code + Text

Ερώτημα Q1: Περιγραφή Λύσης

Ερώτημα:
Βρείτε τα έσοδα του φεστιβάλ, ανά έτος από την πώληση εισιτηρίων, λαμβάνοντας υπόψη όλες τις κατηγορίες εισιτηρίων και παρέχοντας ανάλυση ανά είδος πληρωμής.

1. **Σύνδεση Πινάκων**
   - Ο πίνακας `Ticket` συνδέεται με τον πίνακα `Event` μέσω του `event_id`.
   - Ο πίνακας `Festival` συνδέεται με τον πίνακα `Event` μέσω του `festival_id`. Με αυτόν τον τρόπο καταφέρνουμε να ενώσουμε το κόστος και το είδος των εισιτηρίων με την αντίστοιχη χρονολογία.
2. **Υπολογισμός (`SUM`)**
   - Υπολογίζεται το άθροισμα του κόστους των εισιτηρίων (`SUM(t.cost)`).
3. **Ομαδοποίηση (`GROUP BY`)**
   - Ομαδοποιούμε τα αποτελέσματα πρώτα με `f.year` και στη συνέχεια με `t.payment_method`, ώστε να βλέπουμε ομαδοποιημένα τα αποτελέσματα στην χρονολογία και του τρόπου πληρωμής.
4. **Ταξινόμηση Αποτελεσμάτων (`ORDER BY`)**
   - Τέλος, ταξινομούμε τα αποτελέσματα αρχικά με `f.year` και στη συνέχεια με `t.payment_method`.

```
SELECT f.year,
       t.payment_method,
       SUM(t.cost) AS total_revenue
FROM Ticket t
JOIN Event e ON t.event_id = e.event_id
JOIN Festival f ON e.festival_id = f.festival_id
GROUP BY f.year, t.payment_method
ORDER BY f.year, t.payment_method;
```

Ερώτημα Q2: Περιγραφή Λύσης

Βρείτε όλους τους καλλιτέχνες που ανήκουν σε ένα συγκεκριμένο μουσικό είδος με ένδειξη αν συμμετείχαν σε εκδηλώσεις του φεστιβάλ για το συγκεκριμένο έτος ;

Για την επίλυση του ερωτήματος Q2, ακολουθήσαμε την παρακάτω προσέγγιση:

1. **Επιλογή Πεδίων:** Επιλέγουμε το είδος του καλλιτέχνη (ag.genre από τον πίνακα artist_genre), το id του καλλιτέχνη (a.artist_id από τον πίνακα Artist), το όνομα του καλλιτέχνη (a.name από τον πίνακα Artist) και την ημέρα της παράστασης (e.event_date από τον πίνακα event).

2. **Σύνδεση Πινάκων (JOINs):**
   - Ο πίνακας artist_genre (με ψευδώνυμο ag) συνδέεται με τον πίνακα artist (με ψευδώνυμο a) μέσω του κοινού πεδίου artist_id.
   - Ο πίνακας artist (με ψευδώνυμο a) συνδέεται με τον πίνακα Performance (με ψευδώνυμο p) μέσω του κοινού πεδίου artist_id.
   - Ο πίνακας event (με ψευδώνυμο e) συνδέεται με τον πίνακα Performance (με ψευδώνυμο p) μέσω του κοινού πεδίου event_id.
3. **Φιλτράρισμα Γραμμών (WHERE):**
   - WHERE YEAR(e.event_date) = 2025: Επιλέγονται μόνο οι καλλιτέχνες που εμφανίστηκαν το 2025.
4. **Ταξινόμηση Αποτελεσμάτων (ORDER BY):**
   - Τα αποτελέσματα ταξινομούνται πρωτίστως με βάση το είδος του καλλιτέχνη. Σε περίπτωση κοινού είδους, ταξινομείται βάσει του ονόματος του καλλιτέχνη.

Ο κώδικας SQL που υλοποιεί την παραπάνω λογική είναι ο εξής:

```
SELECT
    ag.genre,
    a.artist_id,
    a.name,
    e.event_date
FROM artist_genre ag
JOIN artist a ON ag.artist_id = a.artist_id
JOIN Performance p ON p.artist_id = a.artist_id
JOIN Event e ON p.event_id = e.event_id
WHERE YEAR(e.event_date) = 2025
ORDER BY
    ag.genre, a.name;
```

Ερώτημα Q03: Περιγραφή Λύσης

Βρείτε ποιοι καλλιτέχνες έχουν εμφανιστεί ως warm up περισσότερες από 2 φορές στο ίδιο φεστιβάλ.

Για την επίλυση του ερωτήματος Q03, ακολούθησα την παρακάτω προσέγγιση:

1. **Επιλογή Πεδίων:** Επιλέγονται το όνομα του καλλιτέχνη (Artist.name από τον πίνακα Artist), την χρονιά του φεστιβάλ(Festival.year από τον πίνακα Festival), και τον αριθμό των εμφανίσεων(performances) που θα χρησιμοποιηθεί έπειτα στο GROUP BY (COUNT(Performance.performance_id) από τον πίνακα Performance).
2. **Σύνδεση Πινάκων (JOINs):**
    - Ο πίνακας Performance συνδέεται με τον πίνακα Event μέσω του κοινού πεδίου event_id.
    - Ο πίνακας Festival συνδέεται με τον πίνακα Event μέσω του κοινού πεδίου festival_id.
    - Ο πίνακας Artist συνδέεται με τον πίνακα Performance μέσω του κοινού πεδίου artist_id.
3. **Φιλτράρισμα Γραμμών (WHERE):**

WHERE Performance.performance_type = 'warm up' : Επιλέγονται μόνο οι εμφανίσεις που είναι τύπου 'warm up'.

4. **Ομαδοποίηση Αποτελεσμάτων (GROUP BY):**

   GROUP BY Artist.name, Festival.year: Τα αποτελέσματα ομαδοποιούνται βάσει του ονόματος του καλλιτέχνη και του έτους του φεστιβάλ. Αυτό είναι απαραίτητο για να λειτουργήσει σωστά η συνάρτηση COUNT() **ΑΝΑ** ερμηνευτή και **ΑΝΑ** φεστιβάλ.

5. **Φιλτράρισμα Ομάδων (HAVING):**

   HAVING COUNT(Performance.performance_id) > 2 : Μετά την ομαδοποίηση, εφαρμόζεται αυτό το φίλτρο για να κρατηθούν μόνο εκείνες οι ομάδες (δηλαδή, οι συνδυασμοί καλλιτέχνη - έτους φεστιβάλ) όπου ο αριθμός των "warm up" εμφανίσεων είναι μεγαλύτερος από 2.

Ο κώδικας SQL που υλοποιεί την παραπάνω λογική είναι ο εξής:

```
SELECT
  Artist.name AS artist_name,
  Festival.year AS festival_year,
  COUNT(Performance.performance_id) AS warm_up_count

FROM Performance
  JOIN Event ON Performance.event_id = Event.event_id
  JOIN Festival ON Event.festival_id = Festival.festival_id
  JOIN Artist ON Performance.artist_id = Artist.artist_id

WHERE
  Performance.performance_type = 'warm up'

GROUP BY
  Artist.name, Festival.year

HAVING
  COUNT(Performance.performance_id) > 2;
```

Ερώτημα Q4: Μέση Βαθμολογίες Καλλιτέχνη

Ερώτημα:

Για κάποιο καλλιτέχνη, βρείτε το μέσο όρο αξιολογήσεων (Ερμηνεία καλλιτεχνών) και εμφάνιση (Συνολική εντύπωση).

Βήματα Επίλυσης:

1. **Σύνδεση Πινάκων**
   - Ο πίνακας `Rating` συνδέεται με τον πίνακα `Performance` μέσω του `performance_id`.
   - Ο πίνακας `Performance` συνδέεται με τον πίνακα `Artist` μέσω του `artist_id`, ώστε να αναγνωριστεί ο καλλιτέχνης που αξιολογήθηκε.
2. **Φιλτράρισμα με `WHERE`**
   - Επιλέγεται μόνο ο καλλιτέχνης με `artist_id = 3`.
3. **Υπολογισμοί (`AVG` και `ROUND`)**
   - Υπολογίζεται ο μέσος όρος των τεσσάρων επιμέρους βαθμολογιών για κάθε εγγραφή, και στη συνέχεια ο μέσος όλων αυτών (`AVG((...)/4.0)`).
   - Υπολογίζεται και ο μέσος όρος της συνολικής βαθμολογίας (`AVG(r.overall_score)`).
   - Και οι δύο τιμές στρογγυλοποιούνται σε 2 δεκαδικά ψηφία με `ROUND(..., 2)`.

Ο SQL κώδικας που υλοποιεί την παραπάνω λογική είναι ο εξής:

```
SELECT
   a.artist_id,
   a.name,
   ROUND(AVG((r.interpretation_score + r.sound_lighting_score +
              r.stage_presence_score + r.organization_score) / 4.0), 2) AS average_rating,
   ROUND(AVG(r.overall_score), 2) AS overall_score
FROM
   Rating r
JOIN
   Performance p ON r.performance_id = p.performance_id
JOIN
   Artist a ON p.artist_id = a.artist_id
WHERE
   a.artist_id = 3
GROUP BY
   a.artist_id, a.name;
```

Ερώτημα Q05: Περιγραφή Λύσης

Βρείτε τους νέους καλλιτέχνες (ηλικία < 30 ετών) που έχουν τις περισσότερες συμμετοχές σε φεστιβάλ;

Για την επίλυση του ερωτήματος Q05, ακολούθησα την παρακάτω προσέγγιση:

1. **Επιλογή Πεδίων:** Επιλέγονται το id του καλλιτέχνη (a.id από τον πίνακα Artist), το όνομα του καλλιτέχνη(a.name από τον πίνακα Artist), και τον αριθμό των διαφορετικών φεστιβάλ (COUNT(DISTINCT f.festival_id) AS festival_count) από τον προσωρινό πίνακα ArtistFestivalCounts.
2. **Σύνδεση Πινάκων (JOINs):**

   Αρχικά, δημιουργώ τον προσωρινό πίνακα YoungArtists για να χρησιμοποιήσω το αποτέλεσμά του στον επόμενο προσωρινό πίνακα ArtistFestivalCounts. Από τον πίνακα Artist επιλέγω τα artist_id και name ώστε να βρω **σήμερα** ποιοι καλλιτέχνες είναι κάτω των 30 χρονών(WHERE DATEDIFF(YEAR, a.date_of_birth, GETDATE()) < 30).

   Μετά, στον πίνακα ArtistFestivalCounts επιλέγω το artist_id από τον πίνακα YoungArtists και τον αριθμό των διαφορετικών φεστιβάλ (COUNT(DISTINCT f.festival_id) AS festival_count). Έτσι θα συνδέσω:

   - τον πίνακα Performance με τον πίνακα YoungArtists μέσω του κοινού πεδίου artist_id.
   - τον πίνακα Event με τον πίνακα Performance μέσω του κοινού πεδίου event_id.
   - τον πίνακα Festival με τον πίνακα Event μέσω του κοινού πεδίου festival_id.

   Επίσης θα κάνω μια ομαδοποίηση αποτελεσμάτων, GROUP BY ya.artist_id: Τα αποτελέσματα ομαδοποιούνται βάσει του artist_id. Αυτό είναι απαραίτητο για να λειτουργήσει σωστά η συνάρτηση COUNT() **ανά** φεστιβάλ.

   Τέλος, θα κάνω JOIN τον πίνακα ArtistFestivalCounts με τον Artist μέσω του κοινού πεδίου artist_id, θα στοιχίσω το festival_count σε φθίνουσα σειρά και θα εμφανίσω μόνο τους τοπ 5 καλλιτέχνες με τις περισσότερες συμμετοχές.

   Ο κώδικας SQL που υλοποιεί την παραπάνω λογική είναι ο εξής:

```
WITH YoungArtists AS (
  SELECT
    a.artist_id,
    a.name,
```

```
    DATEDIFF(YEAR, a.date_of_birth, GETDATE()) AS age
  FROM Artist a
  WHERE DATEDIFF(YEAR, a.date_of_birth, GETDATE()) < 30
),
ArtistFestivalCounts AS (
  SELECT
    ya.artist_id,
    COUNT(DISTINCT f.festival_id) AS festival_count
  FROM YoungArtists ya
  JOIN Performance p ON p.artist_id = ya.artist_id
  JOIN Event e ON p.event_id = e.event_id
  JOIN Festival f ON e.festival_id = f.festival_id
  GROUP BY ya.artist_id
)
SELECT TOP 5
  a.artist_id,
  a.name,
  afc.festival_count
FROM ArtistFestivalCounts afc
JOIN Artist a ON a.artist_id = afc.artist_id
ORDER BY afc.festival_count DESC, a.artist_id;
```

Ερώτημα Q6: Περιγραφή Λύσης

Για κάποιο επισκέπτη, βρείτε τις παραστάσεις που έχει παρακολουθήσει και το μέσο όρο της αξιολόγησης του, ανά παράσταση. (Περιλαμβάνει ανάλυση Query Plan).

Για την επίλυση του ερωτήματος Q6, με στόχο την εύρεση του μέσου όρου αξιολόγησης ανά παράσταση για έναν συγκεκριμένο επισκέπτη (π.χ., `visitor_id = 1`), ακολουθήθηκε την παρακάτω προσέγγιση:

1. **Επιλογή Πεδίων:**
    - `p.performance_id`: Το αναγνωριστικό της κάθε εμφάνισης (performance).
    - `AVG((r.interpretation_score + r.sound_lighting_score + r.stage_presence_score + r.organization_score) / 4.0) AS avg_rating_by_visitor`: Υπολογίζεται ο μέσος όρος

των τεσσάρων επιμέρους κριτηρίων αξιολόγησης για κάθε εμφάνιση. Η διαίρεση με `4.0` εξασφαλίζει αποτέλεσμα κινητής υποδιαστολής. Η συνάρτηση `AVG()` εφαρμόζεται σε αυτόν τον υπολογισμένο μέσο όρο ανά εμφάνιση (λόγω του `GROUP BY`).

2. **Σύνδεση Πινάκων (FROM και JOINs):**
    - `FROM Ticket t`: Το ερώτημα ξεκινά από τον πίνακα `Ticket`.
    - `JOIN Event e ON t.event_id = e.event_id`: Συνδέεται με τον πίνακα `Event` για να βρεθεί το γεγονός στο οποίο αντιστοιχεί το εισιτήριο.
    - `JOIN Performance p ON p.event_id = e.event_id`: Συνδέεται με τον πίνακα `Performance` για να βρεθούν όλες οι εμφανίσεις που ανήκουν στο συγκεκριμένο γεγονός.
    - `LEFT JOIN Rating r ON r.performance_id = p.performance_id AND r.visitor_id = t.visitor_id`: Γίνεται `LEFT JOIN` με τον πίνακα `Rating`. Αυτό είναι σημαντικό για να συμπεριληφθούν όλες οι εμφανίσεις που σχετίζονται με τα εισιτήρια του επισκέπτη για τα γεγονότα που παρακολούθησε, ακόμα κι αν δεν άφησε αξιολόγηση για κάποιες από αυτές. Οι συνθήκες σύνδεσης είναι διπλές: η εμφάνιση πρέπει να ταιριάζει (`r.performance_id = p.performance_id`) ΚΑΙ η αξιολόγηση πρέπει να ανήκει στον συγκεκριμένο επισκέπτη του εισιτηρίου (`r.visitor_id = t.visitor_id`).
3. **Φιλτράρισμα Γραμμών (WHERE):**
    - `WHERE t.visitor_id = 1`: Περιορίζονται τα αποτελέσματα μόνο για τον επισκέπτη με `visitor_id` ίσο με 1.
4. **Ομαδοποίηση Αποτελεσμάτων (GROUP BY):**
    - `GROUP BY p.performance_id`: Τα αποτελέσματα ομαδοποιούνται βάσει του `performance_id`. Αυτό επιτρέπει στη συνάρτηση `AVG()` να υπολογίσει τον μέσο όρο των αξιολογήσεων για κάθε μοναδική εμφάνιση που παρακολούθησε ο συγκεκριμένος επισκέπτης.

Ο βασικός κώδικας SQL που υλοποιεί την παραπάνω λογική είναι ο εξής:

```
SELECT
    p.performance_id,
    AVG(
        (r.interpretation_score + r.sound_lighting_score + r.stage_presence_score + r.orga
    ) AS avg_rating_by_visitor
FROM Ticket t
JOIN Event e ON t.event_id = e.event_id
JOIN Performance p ON p.event_id = e.event_id
LEFT JOIN Rating r ON r.performance_id = p.performance_id AND r.visitor_id = t.visitor_id
```

```
WHERE t.visitor_id = 1
GROUP BY p.performance_id;
```

Ερώτημα Q7: Περιγραφή Λύσης

Βρείτε ποιο φεστιβάλ είχε τον χαμηλότερο μέσο όρο εμπειρίας τεχνικού προσωπικού;

Για την επίλυση του ερωτήματος Q7, ακολουθήσαμε την παρακάτω προσέγγιση:

1. **Επιλογή Πεδίων:** Επιλέγουμε το id του festival(f.festival_id από τον πίνακα festival)και τον συνδυασμός της εντολής AVG(),ROUND(..., 2) με τη δομή CASE ώστε να υπολογιστεί ο μέσος όρος εμπειρίας του προσωπικού ανά φεστιβάλ, μετατρέποντας τις κατηγορικές τιμές εμπειρίας (π.χ. 'μέσος', 'έμπειρος') σε αριθμητικές τιμές από το 1 έως το 5 και το αποτέλεσμα να εμφανίζεται με 2 δεκαδικά ψηφία
2. **Σύνδεση Πινάκων (JOINs):**
    - Ο πίνακας festival (με ψευδώνυμο f) συνδέεται με τον πίνακα event (με ψευδώνυμο e) μέσω του κοινού πεδίου festival_id.
    - Ο πίνακας event (με ψευδώνυμο e) συνδέεται με τον πίνακα event_staff (με ψευδώνυμο es) μέσω του κοινού πεδίου event_id.
    - Ο πίνακας event_staff (με ψευδώνυμο es) συνδέεται με τον πίνακα staff (με ψευδώνυμο s) μέσω του κοινού πεδίου staff_id.
3. **Ομαδοποίηση Αποτελεσμάτων (GROUP BY):**
    - Ομαδοποιούμε τα αποτελέσματα με βάση το id του festival
4. **Ταξινόμηση Αποτελεσμάτων (ORDER BY):**
    - Τα αποτελέσματα ταξινομούνται με βάση το μέσο όρο εμπειρίας τεχνικού προσωπικού σε αύξουσα σειρά.
5. **Επιλογή του πρώτου αποτελέσματος**(SELECT TOP 1)

Ο κώδικας SQL που υλοποιεί την παραπάνω λογική είναι ο εξής:

```
SELECT TOP 1
        f.festival_id,
        ROUND(AVG(
```

```
            CASE s.experience_level
                WHEN 'ειδικευόμενος' THEN 1
                WHEN 'αρχάριος' THEN 2
                WHEN 'μέσος' THEN 3
                WHEN 'έμπειρος' THEN 4
                WHEN 'πολύ έμπειρος' THEN 5
            END
        ), 2) AS avg_experience_score
    FROM Festival f
    JOIN Event e ON f.festival_id = e.festival_id
    JOIN Event_Staff es ON e.event_id = es.event_id
    JOIN Staff s ON es.staff_id = s.staff_id
    GROUP BY f.festival_id
    ORDER BY avg_experience_score ASC;
```

Ερώτημα Q8: Περιγραφή Λύσης

Βρείτε το προσωπικό υποστήριξης που δεν έχει προγραμματισμένη εργασία σε συγκεκριμένη ημερομηνία;

Για την επίλυση του ερωτήματος Q8, ακολουθήσαμε την παρακάτω προσέγγιση:

1. **Δημιουργεία όλων των πιθανών συνδυασμών αναθέσεις μεταξύ εκδηλώσεων και μελών υποστηρικτικού προσωπικού(All_Possible_Combinations).**
2. **Επιλογή Πεδίων:**
    - Επιλέγουμε τη ημέρα του event(e.event_date από τον πίνακα event) και το id του προσωπικού(es.staff_id από τον πίνακα Event_Staff).
3. **Σύνδεση Πινάκων (JOINs):**
    - Ο πίνακας event (με ψευδώνυμο e) συνδέεται με τον πίνακα event_staff (με ψευδώνυμο es) με χρήση CROSS JOIN(Όλα τα e.event_date με όλα es.staff_id).
4. **Φιλτράρισμα Γραμμών (WHERE):**
    - es.staff_category = 'auxiliary': Επιλέγονται μόνο τα μέλοι του υποστιρικτικού προσωπικού.

5. **Επιστροφή των πραγματικών αναθέσεων βοηθητικού προσωπικού ανά ημερομηνία event(Assigned_Dates).**

6. **Επιλογή Πεδίων:**
    - Επιλέγουμε τη ημέρα του event(e.event_date από τον πίνακα event) και το id του προσωπικού(es.staff_id από τον πίνακα Event_Staff).
7. **Σύνδεση Πινάκων (JOINs):**
    - Ο πίνακας event (με ψευδώνυμο e) συνδέεται με τον πίνακα event_staff (με ψευδώνυμο es) μέσω του κοινού πεδίου event_id.
8. **Φιλτράρισμα Γραμμών (WHERE):**
    - es.staff_category = 'auxiliary': Επιλέγονται μόνο τα μέλοι του υποστιρικτικού προσωπικού.
9. **Τελική Επιλογη**

10. **Επιλογή Πεδίων:** * Επιλέγουμε τη ημέρα του event(event_date από τον πίνακα All_Possible_Combinations), το id του προσωπικού(apc.staff_id από τον πίνακα ll_Possible_Combinations και το όνομα του πρωσοπικού(name από τον πίνακα Staff).
11. **Σύνδεση Πινάκων (JOINs):** * Ο πίνακας staff (με ψευδώνυμο s) συνδέεται με τον πίνακα All_Possible_Combinations (με ψευδώνυμο apc) μέσω του κοινού πεδίου staff_id. * Ο πίνακας All_Possible_Combinations (με ψευδώνυμο apc) συνδέεται με τον πίνακα Assigned_Dates(με ψευδώνυμο ad) με χρήση LEFT EXCLUDING JOIN(Επιλέγει όλα τα στοιχεία του πίνακα All_Possible_Combinations που δεν είναι κοινά με τον πίνακα Assigned_Dates) μέσω των κοινών πεδίων staff_id και event_date.
12. **Ταξινόμηση Αποτελεσμάτων (ORDER BY):** * Τα αποτελέσματα ταξινομούνται πρωτίστως με βάση τη ημερομηνία του event. Σε περίπτωση κοινού event, ταξινομείται βάσει το id του staff.
Ο κώδικας SQL που υλοποιεί την παραπάνω λογική είναι ο εξής:

```
WITH All_Possible_Combinations AS (
    SELECT
        e.event_date,
        es.staff_id
    FROM Event e
    CROSS JOIN Event_Staff es
    WHERE es.staff_category = 'auxiliary'
),
Assigned_Dates AS (
    SELECT
        e.event_date,
        es.staff_id
```

```
    FROM Event e
    JOIN Event_Staff es ON e.event_id = es.event_id
    WHERE es.staff_category = 'auxiliary'
)
SELECT
    apc.event_date,
    apc.staff_id,
    s.name
FROM Staff s
JOIN All_Possible_Combinations apc ON s.staff_id = apc.staff_id
LEFT JOIN Assigned_Dates ad
    ON apc.staff_id = ad.staff_id AND apc.event_date = ad.event_date
WHERE ad.staff_id IS NULL
ORDER BY apc.event_date, apc.staff_id;
```

Ερώτημα Q9: Περιγραφή Λύσης

Βρείτε ποιοι επισκέπτες έχουν παρακολουθήσει τον ίδιο αριθμό παραστάσεων σε διάστημα ενός έτους με περισσότερες από 3 παρακολουθήσεις;

Για την επίλυση του ερωτήματος Q9, ακολουθήσαμε την παρακάτω προσέγγιση:

1. **Υπολογισμός ξεχωριστών παρακολουθήσεων σε διάστημα ενός έτους(yearly_counts)**
2. **Επιλογή Πεδίων:**
    - πιλέγουμε το ID του επισκέπτη (t.visitor_id από τον πίνακα Ticket), τον χρόνο του event (YEAR(e.event_date) από τον πίνακα Event) και το σύνολο των διαφορετικών ID των εισιτηρίων (COUNT(DISTINCT t.event_id) AS num_events από τον πίνακα Ticket).
3. **Σύνδεση Πινάκων (JOINs):**
    - Ο πίνακας Ticket (με ψευδώνυμο t) συνδέεται με τον πίνακα Event (με ψευδώνυμο e) μέσω του κοινού πεδίου event_id.
4. **Ταξινόμηση Αποτελεσμάτων (ORDER BY):**
    - Τα αποτελέσματα ταξινομούνται πρωτίστως με βάση το id του επισκέπτη. Σε περίπτωση κοινού επισκέπτη, ταξινομείται βάσει του id του έτους.
5. **Φιλτράρισμα παρακολουθήσεων(filtered_counts)**
6. **Φιλτράρισμα Γραμμών (WHERE):**

- num_events >= 3 Επιλέγονται μόνο όταν ο αριθμός των event είναι μεγαλύτερος από 3.

7. **Βρίσκουμε τους επισκέπτες έχουν παρακολουθήσει τον ίδιο αριθμό παραστάσεων σε διάστημα ενός έτους με περισσότερες από 3 παρακολουθήσεις**
8. **Επιλογή Πεδίων:**
   - πιλέγουμε το ID του επισκέπτη (a.visitor_id από τον πίνακα filtered_counts), τον χρόνο του event (a.year από τον πίνακα filtered_counts) και το σύνολο των διαφορετικών eventt(a.num_events από τον πίνακα filtered_counts).
9. ****Σύνδεση Πινάκων (JOINs):**
   - Ο πίνακας filtered_counts (με ψευδώνυμο a) συνδέεται με τον πίνακα filtered_counts (με ψευδώνυμο b) μέσω του κοινού πεδίου num_events και κοινού πεδίου visitor_id.
10. **Ταξινόμηση Αποτελεσμάτων (ORDER BY):**
    - Τα αποτελέσματα ταξινομούνται πρωτίστως με βάση του έτους. Σε περίπτωση κοινού επισκέπτη, ταξινομείται βάσει τον αριθμό events. Σε περίπτωση κοινού αριθμού events, ταξινομείται βάσει το id του επισκέπτη

Ο κώδικας SQL που υλοποιεί την παραπάνω λογική είναι ο εξής:

```
WITH yearly_counts AS (
   SELECT
       t.visitor_id,
       YEAR(e.event_date) AS year,
       COUNT(DISTINCT t.event_id) AS num_events
   FROM Ticket t
   JOIN Event e ON t.event_id = e.event_id
   GROUP BY t.visitor_id, YEAR(e.event_date)
),
filtered_counts AS (
   SELECT *
   FROM yearly_counts
   WHERE num_events >= 3
)
SELECT DISTINCT a.visitor_id, a.year, a.num_events
FROM filtered_counts a
JOIN filtered_counts b
  ON a.year = b.year
 AND a.num_events = b.num_events
```

```
  AND a.visitor_id <> b.visitor_id
ORDER BY a.year, a.num_events, a.visitor_id;
```

Ερώτημα Q10: Περιγραφή Λύσης

Πολλοί καλλιτέχνες καλύπτουν περισσότερα από ένα μουσικά είδη. Ανάμεσα σε ζεύγη πεδίων (π.χ. ροκ, τζαζ) που είναι κοινά στους καλλιτέχνες, βρείτε τα 3 κορυφαία (top-3) ζεύγη που εμφανίστηκαν σε φεστιβάλ.

Για την επίλυση του ερωτήματος Q10, ακολούθησα την παρακάτω προσέγγιση:

1. **Επιλογή Πεδίων:** Επιλέγονται το 1ο μουσικό είδος(genre1), το 2ο μουσικό είδος(genre2), και τον αριθμό των φεστιβάλ(festivals_count) που αναφέρεται στα διαφορετικά φεστιβάλ όπου εμφανίζεται ένα ζεύγος πεδίου.
2. **Σύνδεση Πινάκων (JOINs):**

   Αρχικά, δημιουργώ τον προσωρινό πίνακα GenrePairs. Από τον πίνακα Artist_Genre επιλέγω τα artist_id και 2 genre(genre1 και genre2). Θα κάνω Self Join στον πίνακα Artist_Genre μέσω του κοινού πεδίου artist_id. Έτσι θα πάρω έναν πίνακα όπου σε καθε γραμμή θα έχω τον καλλιτέχνη και το ζεύγος μουσικών ειδών του.

   Εξήγηση του AND ag1.genre < ag2.genre: Αυτό γίνεται για να αποφύγουμε το ίδιο ζεύγος 2 φορές(πχ. (Rock, Jazz) (Jazz, Rock)). Ελέγχει αν το 1ο γράμμα του ag1.genre έρχεται πιο πριν στην αλφαβήτα απο το ag2.genre και τότε το συμπεριλαμβάνει στο JOIN(πχ. το 'α' είναι πριν το 'β'), αλλίως το προσπερνάει.

   Μετά, δημιουργώ τον προσωρινό πίνακα GenreFestivalAppearances. Από τον πίνακα GenrePairs επιλέγω τα genre1 και genre2. Από τον πίνακα Festival επιλέγω τα festival_id.

   - Ο πίνακας Performance συνδέεται με τον πίνακα GenrePairs μέσω του κοινού πεδίου artist_id.
   - Ο πίνακας Event συνδέεται με τον πίνακα Performance μέσω του κοινού πεδίου event_id.
   - Ο πίνακας Festival συνδέεται με τον πίνακα Event μέσω του κοινού πεδίου festival_id.

   Έπειτα δημιουργώ τον προσωρινό πίνακα GenrePairFestivalCounts. Από τον πίνακα GenreFestivalAppearances επιλέγω τα genre1, genre2 και COUNT(DISTINCT festival_id)

όπου μετράει τον αριθμό μόνο των διαφορετικών φεστιβάλ.

Στο τελικό SELECT επιλέγω genre1, genre2, festivals_count και θα στοιχίσω το festival_count σε φθίνουσα σειρά και θα εμφανίσω μόνο τους τοπ 3 μουσικά είδη.

3. **Ομαδοποίηση Αποτελεσμάτων (GROUP BY):**

   GROUP BY genre1, genre2: Αυτό είναι απαραίτητο για να λειτουργήσει σωστά η συνάρτηση COUNT() που μετράει τον αριθμό των φεστιβάλ **ΑΝΑ** ζεύγος ειδών.

Ο κώδικας SQL που υλοποιεί την παραπάνω λογική είναι ο εξής:

```
WITH GenrePairs AS (
  SELECT
      ag1.artist_id,
      ag1.genre AS genre1,
      ag2.genre AS genre2
  FROM Artist_Genre ag1
  JOIN Artist_Genre ag2
      ON ag1.artist_id = ag2.artist_id
      AND ag1.genre < ag2.genre
),
GenreFestivalAppearances AS (
  SELECT DISTINCT
      gp.genre1,
      gp.genre2,
      f.festival_id
  FROM GenrePairs gp
  JOIN Performance p ON gp.artist_id = p.artist_id
  JOIN Event e ON p.event_id = e.event_id
  JOIN Festival f ON e.festival_id = f.festival_id
),
GenrePairFestivalCounts AS (
  SELECT
      genre1,
      genre2,
      COUNT(DISTINCT festival_id) AS festivals_count
  FROM GenreFestivalAppearances
  GROUP BY genre1, genre2
)
SELECT TOP 3
  genre1,
  genre2,
  festivals_count
```

```
    FROM GenrePairFestivalCounts
    ORDER BY festivals_count DESC;
```

Ερώτημα Q11: Περιγραφή Λύσης

Βρείτε όλους τους καλλιτέχνες που συμμετείχαν τουλάχιστον 5 λιγότερες φορές από τον καλλιτέχνη με τις περισσότερες συμμετοχές σε φεστιβάλ.

Για την επίλυση του ερωτήματος Q11, ακολούθησα την παρακάτω προσέγγιση:

Αρχικά δημιουργώ τον προσωρινό πίνακα ArtistParticipation ώστε να βρω σε πόσα φεστιβάλ έχει βρεθεί ο κάθε καλλιτέχνης. Στο πεδίο COUNT(DISTINCT Festival.festival_id) AS total_participations επιλέγω το DISTINCT διότι σε ένα φεστιβάλ ένας καλλιτέχνης μπορεί να έχει πολλές εμφανίσεις.

Μετά, στον προσωρινό πίνακα MaxParticipation βρίσκω τον καλλιτέχνη με τις περισσότερες εμφανίσεις με την χρήση του MAX(total_participations).

Τέλος, χρησιμοποιώ τα πεδία artist_id, artist_name, total_participations από τον πίνακα ArtistParticipation. Με την χρήση του JOIN MaxParticipation mp ON 1 = 1, κάνω cross join. Έτσι ενώνω όλες τις γραμμές του ArtistParticipation με την γραμμή(τοπ καλλιτέχνη) MaxParticipation.

Φιλτράρισμα Γραμμών (WHERE): WHERE ap.total_participations <= mp.max_participations - 5; Επιστρέφει όλους τους καλλιτέχνες που συμμετείχαν τουλάχιστον 5 λιγότερες φορές από τον καλλιτέχνη με τις περισσότερες συμμετοχές σε φεστιβάλ

Ο κώδικας SQL που υλοποιεί την παραπάνω λογική είναι ο εξής:

```
 WITH ArtistParticipation AS (
    SELECT
        Artist.artist_id,
        Artist.name AS artist_name,
        COUNT(DISTINCT Festival.festival_id) AS total_participations
    FROM Performance
    JOIN Artist ON Artist.artist_id = Performance.artist_id
    JOIN Event ON Performance.event_id = Event.event_id
    JOIN Festival ON Event.festival_id = Festival.festival_id
```

```
    GROUP BY Artist.artist_id, Artist.name
),
MaxParticipation AS (
    SELECT MAX(total_participations) AS max_participations
    FROM ArtistParticipation
)
SELECT
    ap.artist_id,
    ap.artist_name,
    ap.total_participations
FROM ArtistParticipation ap
JOIN MaxParticipation mp ON 1 = 1
WHERE ap.total_participations <= mp.max_participations - 5;
```

Ερώτημα Q12: Υπολογισμός Προσωπικού για Εκδηλώσεις

Ερώτημα:
Βρείτε το προσωπικό που απαιτείται για κάθε ημέρα του φεστιβάλ, παρέχοντας ανάλυση ανά κατηγορία (τεχνικό προσωπικό ασφαλείας, βοηθητικό προσωπικό);

Βήματα Επίλυσης:

1. **Σύνολο Εισιτηρίων Ανά Εκδήλωση (`total_visits`)**
   Υπολογίζουμε το πλήθος των εισιτηρίων (`ticket_id`) που έχουν εκδοθεί για κάθε εκδήλωση (`event_id`).
2. **Υπολογισμός Προσωπικού Ασφαλείας (`Security_Personnel`)**
   Για κάθε εκδήλωση, ο αριθμός ατόμων ασφαλείας προκύπτει ως το **5% του πλήθους εισιτηρίων**, στρογγυλοποιημένο προς τα πάνω (`CEILING`).
3. **Υπολογισμός Βοηθητικού Προσωπικού (`auxiliary_Personnel`)**
   Για κάθε εκδήλωση, ο αριθμός βοηθητικού προσωπικού είναι το **2% των εισιτηρίων**, επίσης στρογγυλοποιημένο προς τα πάνω.
4. **Καθορισμός Τεχνικού Προσωπικού**
   Ο αριθμός τεχνικού προσωπικού θεωρείται σταθερός: **20 άτομα ανά εκδήλωση**.
5. **Τελική Επιλογή και Ταξινόμηση**
   Επιλέγονται τα `event_id`, ο αριθμός κάθε κατηγορίας προσωπικού και ταξινομούνται με

βάση το `event_id`.

```
WITH total_visits AS(
SELECT
e.event_id,
COUNT(t.ticket_id) AS total_tickets
FROM Event e
JOIN Ticket t on e.event_id = t.event_id
GROUP BY e.event_id
),
Security_Personnel AS (
    SELECT
        tv.event_id,
        CEILING(tv.total_tickets * 0.05) AS num_of_security
    FROM total_visits tv
),
auxiliary_Personnel AS (
    SELECT
        tv.event_id,
        CEILING(tv.total_tickets * 0.02) AS num_of_auxiliary
    FROM total_visits tv
)
SELECT
    sp.event_id,
    sp.num_of_security,
    sec.num_of_auxiliary,
    20 AS num_of_technical
FROM Security_Personnel sp
JOIN auxiliary_Personnel sec ON sp.event_id = sec.event_id
ORDER BY sp.event_id;
```

Ερώτημα Q13: Καλλιτέχνες με Συμμετοχές σε Τουλάχιστον 3 Διαφορετικές Ηπείρους

Ερώτημα:
Να βρεθούν οι καλλιτέχνες που έχουν συμμετάσχει σε φεστιβάλ τα οποία έλαβαν χώρα σε **τουλάχιστον 3 διαφορετικές ηπείρους**.

Βήματα Επίλυσης:

1. **Σύνδεση Πινάκων**
   Πραγματοποιούνται τα εξής `JOIN`:
   - `Festival` με `Location` για να βρούμε την ήπειρο κάθε φεστιβάλ.
   - `Festival` με `Event`, και `Event` με `Performance`, ώστε να συσχετίσουμε τα φεστιβάλ με τις εμφανίσεις των καλλιτεχνών.
   - `Performance` με `Artist` για να καταλήξουμε στο όνομα του καλλιτέχνη.
2. **Ομαδοποίηση (`GROUP BY`)**
   Ομαδοποιούμε τα αποτελέσματα με βάση `artist_id` ώστε να μετρήσουμε για κάθε καλλιτέχνη σε πόσες διαφορετικές ηπείρους έχει συμμετάσχει.
3. **Φιλτράρισμα (`WHERE`)**
   Κρατάμε μόνο τους καλλιτέχνες που έχουν συμμετάσχει σε φεστιβάλ σε **τουλάχιστον 3 διαφορετικές ηπείρους**.

Ο κώδικας SQL που υλοποιεί την παραπάνω λογική είναι ο εξής:

```
WITH Different_Continents AS (
   SELECT
       a.artist_id,
       a.name,
       COUNT(DISTINCT l.continent) AS distinct_continents
   FROM festival f
   JOIN location l ON f.location_id = l.location_id
   JOIN event e ON f.festival_id = e.festival_id
   JOIN performance p ON p.event_id = e.event_id
   JOIN artist a ON p.artist_id = a.artist_id
   GROUP BY a.artist_id
),
filtered_counts AS (
   SELECT *
   FROM Different_Continents
   WHERE distinct_continents >= 3
)
SELECT * FROM filtered_counts;
```

Ερώτημα Q14: Περιγραφή Λύσης

Βρείτε ποια μουσικά είδη είχαν τον ίδιο αριθμό εμφανίσεων σε δύο συνεχόμενες χρονιές με τουλάχιστον 3 εμφανίσεις ανά έτος;

Για την επίλυση του ερωτήματος Q10, ακολούθησα την παρακάτω προσέγγιση:

Αρχικά δημιούργησα τον προσωρινό πίνακα GenreYearCount με σκοπό να βρω τα μουσικά είδη που είχαν πάνω απο τρεις εμφανίσεις σε ένα φεστιβάλ. Θα κάνω τα εξής JOIN:

- Ο πίνακας Performance συνδέεται με τον πίνακα Artist μέσω του κοινού πεδίου artist_id.
- Ο πίνακας Artist_Genre συνδέεται με τον πίνακα Artist μέσω του κοινού πεδίου artist_id.
- Ο πίνακας Event συνδέεται με τον πίνακα Performance μέσω του κοινού πεδίου event_id.
- Ο πίνακας Festival συνδέεται με τον πίνακα Event μέσω του κοινού πεδίου festival_id.

Ομαδοποίηση Αποτελεσμάτων (GROUP BY):

GROUP BY ag.genre, f.year: Τα αποτελέσματα ομαδοποιούνται βάσει του ονόματος του καλλιτέχνη και του έτους του φεστιβάλ. Αυτό είναι απαραίτητο για να λειτουργήσει σωστά η συνάρτηση COUNT() **ΑΝΑ** είδος και **ΑΝΑ** χρονιά φεστιβάλ.

Φιλτράρισμα Ομάδων (HAVING):

HAVING COUNT(p.performance_id) >= 3: Μετά την ομαδοποίηση, εφαρμόζεται αυτό το φίλτρο για να κρατηθούν μόνο εκείνα τα μουσικά είδη όπου συνολικά είχαν πάνω από 3 εμφανίσεις.

Μετά δημιουργώ τον προσωρινό πίνακα MatchingGenres. Κάνω SELF-JOIN τον πίνακα GenreYearCount με την προυπόθεση οτι ελέγχω δυο συνεχόμενες χρονιές(AND g2.year = g1.year + 1) και οι εμφανίσεις και στις δυο χρονιές είναι ίσες(AND g1.appearances = g2.appearances).

Τέλος, εμφανίζω το μουσικό είδος, τις 2 συνεχόμενες χρονιές και πόσες εμφανίσεις είχε(που θα είναι πάνω απο 3).

Ο κώδικας SQL που υλοποιεί την παραπάνω λογική είναι ο εξής:

```
WITH GenreYearCount AS (
  SELECT
      ag.genre,
      f.year,
      COUNT(p.performance_id) AS appearances
  FROM Performance p
  JOIN Artist a ON p.artist_id = a.artist_id
  JOIN Artist_Genre ag ON a.artist_id = ag.artist_id
```

```
  JOIN Event e ON p.event_id = e.event_id
  JOIN Festival f ON e.festival_id = f.festival_id
  GROUP BY ag.genre, f.year
  HAVING COUNT(p.performance_id) >= 3
),
MatchingGenres AS (
  SELECT
      g1.genre,
      g1.year AS year1,
      g2.year AS year2,
      g1.appearances
  FROM GenreYearCount g1
  JOIN GenreYearCount g2
      ON g1.genre = g2.genre
      AND g2.year = g1.year + 1
      AND g1.appearances = g2.appearances
)
SELECT *
FROM MatchingGenres
ORDER BY genre, year1;
```

Ερώτημα Q15: Περιγραφή Λύσης

Βρείτε τους top-5 επισκέπτες που έχουν δώσει συνολικά την υψηλότερη βαθμολόγηση σε ένα καλλιτέχνη. (όνομα επισκέπτη, όνομα καλλιτέχνη και συνολικό σκορ βαθμολόγησης).

Για την επίλυση του ερωτήματος Q15, ακολούθησα την παρακάτω προσέγγιση:

1. **Επιλογή Πεδίων:** Επιλέγονται το όνομα του επισκέπτη (`v.first_name` από τον πίνακα `Visitor`), το όνομα του καλλιτέχνη (`a.name` από τον πίνακα `Artist`), και η συνολική βαθμολογία (`r.overall_score` από τον πίνακα `Rating`).
2. **Σύνδεση Πινάκων (JOINs):**
    - Ο πίνακας `Rating` (με ψευδώνυμο `r`) συνδέεται με τον πίνακα `Visitor` (με ψευδώνυμο `v`) μέσω του κοινού πεδίου `visitor_id`.
    - Ο πίνακας `Rating` (`r`) συνδέεται με τον πίνακα `Performance` (με ψευδώνυμο `p`) μέσω του κοινού πεδίου `performance_id`.
    - Ο πίνακας `Performance` (`p`) συνδέεται με τον πίνακα `Artist` (με ψευδώνυμο `a`) μέσω του κοινού πεδίου `artist_id`. Αυτές οι συνδέσεις επιτρέπουν τη συλλογή των

απαραίτητων πληροφοριών (όνομα επισκέπτη, όνομα καλλιτέχνη, βαθμολογία) από τους αντίστοιχους πίνακες για κάθε αξιολόγηση.

3. **Ταξινόμηση Αποτελεσμάτων (ORDER BY):**
    - Τα αποτελέσματα ταξινομούνται πρωτίστως κατά φθίνουσα σειρά της συνολικής βαθμολογίας (`r.overall_score DESC`). Αυτό φέρνει τις υψηλότερες βαθμολογίες στην κορυφή.
    - Σε περίπτωση ισοβαθμίας στη συνολική βαθμολογία, τα αποτελέσματα ταξινομούνται δευτερευόντως αλφαβητικά βάσει του πρώτου ονόματος του επισκέπτη (`v.first_name ASC` - το `ASC` είναι προεπιλογή και μπορεί να παραλειφθεί).
4. **Περιορισμός Αριθμού Αποτελεσμάτων (OFFSET FETCH):**
    - Η εντολή `OFFSET 0 ROWS FETCH NEXT 5 ROWS ONLY` χρησιμοποιείται για να παραλειφθούν οι πρώτες 0 γραμμές (δηλαδή, καμία γραμμή δεν παραλείπεται από την αρχή) και στη συνέχεια να επιλεγούν οι επόμενες 5 γραμμές. Αυτό ουσιαστικά επιστρέφει τις 5 κορυφαίες εγγραφές μετά την ταξινόμηση.

Ο κώδικας SQL που υλοποιεί την παραπάνω λογική είναι ο εξής:

```
SELECT
    v.first_name,
    a.name,
    r.overall_score
    FROM Rating r
    JOIN Visitor v ON r.visitor_id = v.visitor_id
    JOIN Performance p ON r.performance_id = p.performance_id
    JOIN Artist a ON p.artist_id = a.artist_id
    ORDER BY r.overall_score DESC, v.first_name
    OFFSET 0 ROWS FETCH NEXT 5 ROWS ONLY;
```

Could not connect to the reCAPTCHA service. Please check your internet connection and reload to get a reCAPTCHA challenge.