



## ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ

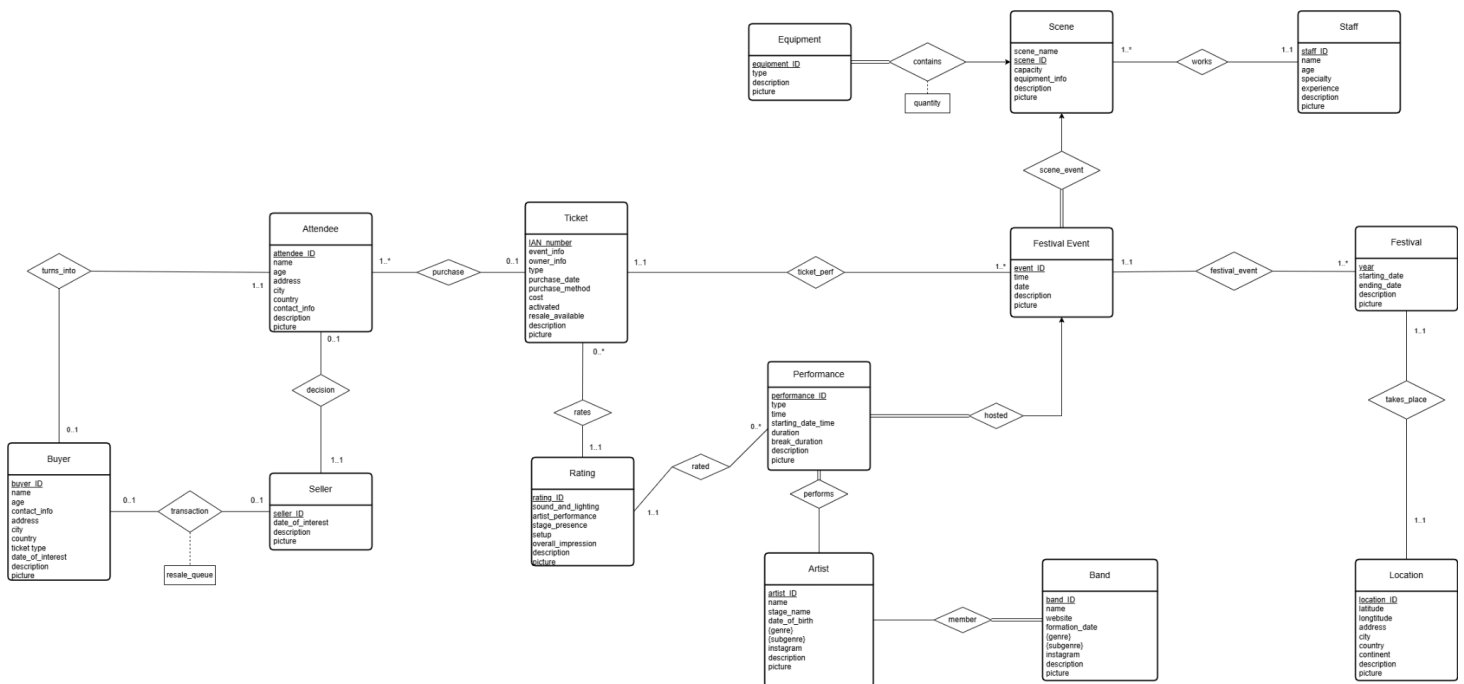
### ΣΧΟΛΗ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ

#### 6ο Εξάμηνο, Βάσεις Δεδομένων Εξαμηνιαία Εργασία

##### Επιμέλεια:

- Παλαιοκώστας Μάριος Ιωάννης, 03122205
- Παναγιωταράκος Αλέξιος, 03122200
- Ρούσσος Φίλιππος Νικόλαος, 03122118

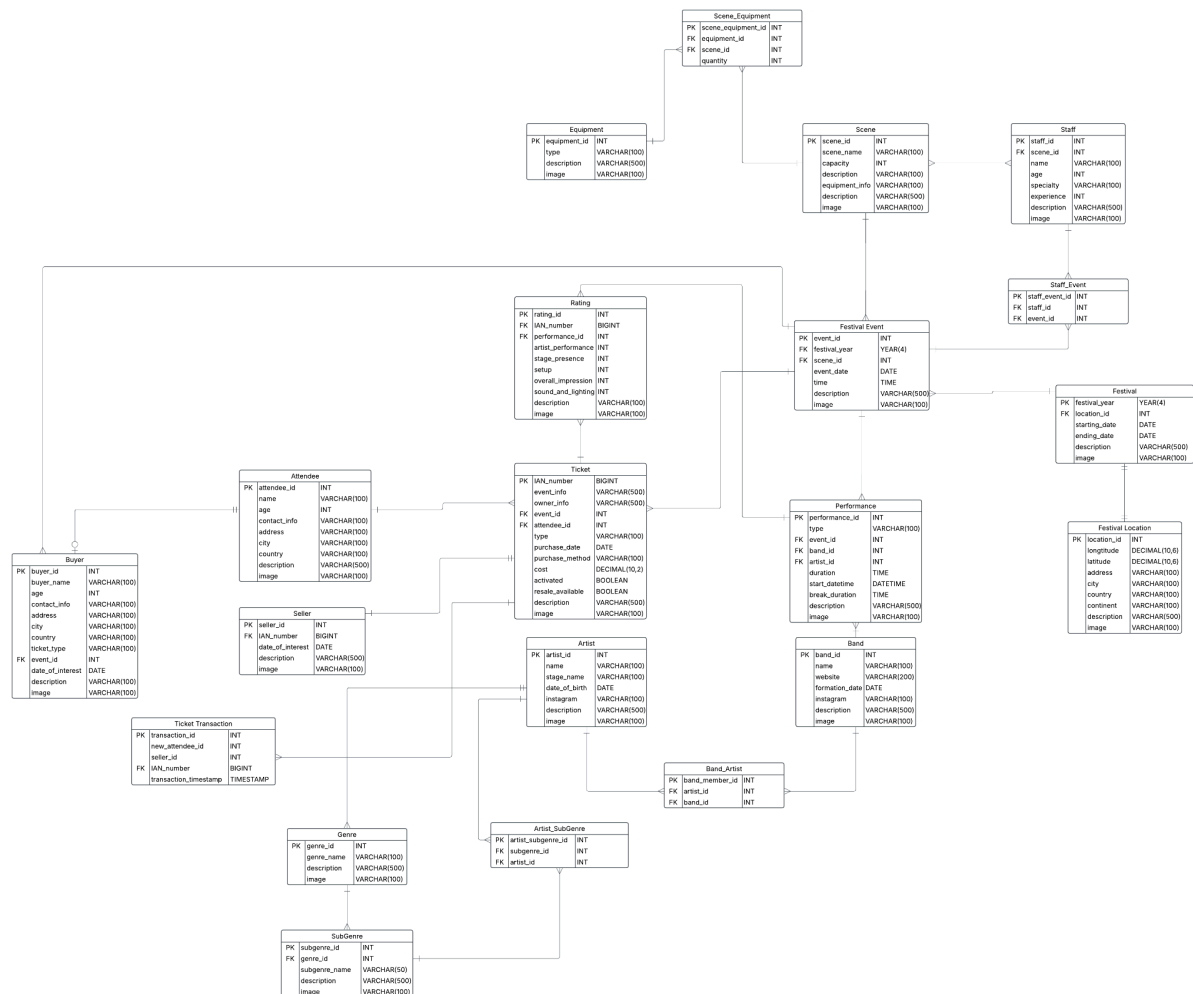
##### A.1 ER Diagram



Σχεδιάσαμε το ER diagram μέσω της εφαρμογής [draw.io](https://draw.io) το διάγραμμα επίσης βρίσκεται στο αρχείο pdf ER\_music\_festival.

Τα Primary Keys είναι τα υπογραμμισμένα στοιχεία σε κάθε table. Τα στοιχεία genre, subgenre που εμφανίζονται στα band, artist έχουν τοποθετηθεί σε άγκιστρα καθώς μπορεί μια μπάντα ή ένας καλλιτέχνης να παίζουν πολλά είδη/υποείδη. Για την σύνδεση του εξοπλισμού με την σκηνή υλοποιήσαμε ένα junction table που επίσης περιέχει την ποσότητα του εξοπλισμού. Για την υλοποίηση του transaction που συνδέει buyer,seller , δημιουργήσαμε μία ξεχωριστή οντότητα ticket\_transaction στο sql schema για να καταγράφεται η αγοραπωλησία των εισιτηρίων μέσω του procedure ProcessAllResaleTransactions.

## A.2 Relational Diagram



Σχεδιάσαμε το relational diagram μέσω της εφαρμογής Lucidchart το διάγραμμα επίσης βρίσκεται στο pdf Relational Diagram Music Festival.

Σε κάθε many-to-many relationship υλοποιήσαμε junction tables για να μετατρέψουμε κάθε many-to-many σε δύο one-to-many (κάθε εγγραφή του ενός πίνακα να σχετίζεται με πολλές εγγραφές του άλλου πίνακα και αντίστροφα), έτσι ο ενδιαμέσος πίνακας παίρνει τα primary keys των entities που συνδέει δημιουργώντας ένα καινούργιο primary key για κάθε πιθανό συνδυασμό τους.

Όσον αφορά το table buyer τα στοιχεία που περιέχει είναι όμοια με του attendee καθώς όταν καλείται το procedure του resale queue οι αγοραστές που βρίσκουν το επιθυμητό τους εισιτήριο εισάγονται στο table attendee ενώ διαγράφονται από το table των buyers.

Το table ticket\_transaction αποτελεί πίνακα για την καταγραφή αγορών-πωλήσεων. Το attribute new\_attendee\_id είναι το attendee\_id που έχει λάβει ο εκάστοτε αγοραστής.

### A.2.1 Περιορισμοί

Για ευκολία, στα junction tables και για άλλες ορισμένες οντότητες έχουμε επιλέξει να ορίσουμε ως primary key ένα νέο attribute {table\_name}\_id αντί για την ένωση πολλαπλών foreign keys.

Έχουμε εισάγει τους εξής περιορισμούς στα foreign keys:

- ON DELETE RESTRICT ON UPDATE NO ACTION

Το ON DELETE RESTRICT απαγορεύει την διαγραφή ενός στοιχείου του πίνακα “γονέα” αν υπάρχουν εξαρτώμενα στοιχεία στον τρέχον πίνακα.

Το ON UPDATE NO ACTION δεν επιτρέπει την ενημέρωση της τιμής στον πίνακα “γονέα”.

Από την εκφώνηση προκύπτουν ορισμένα constraints που συμπεριλάβαμε με την μορφή CHECKS στα αντίστοιχα tables. Συγκεκριμένα:

- Στο scene CHECK(capacity > 0) που εξασφαλίζει μια υπαρκτή χωρητικότητα της σκηνής
- Στο festival CHECK (ending\_date >= starting\_date) για έγκυρες ημερομηνίες διεξαγωγής του φεστιβάλ
- Στο performance CHECK (duration < 180) για διάρκεια εμφάνισης μικρότερη των 3 ωρών, CHECK (break\_duration >= 5 AND break\_duration <= 30) για διάρκεια διαλείμματος μεταξύ των εμφανίσεων 5-30 λεπτά, CHECK ((artist\_id IS NOT NULL AND band\_id IS NULL) OR (artist\_id IS NULL AND band\_id IS NOT NULL)) για να ελέγχουμε ότι εμφανίζεται μπάντα ή σόλο καλλιτέχνης.
- Στο ticket CHECK(activated = FALSE or resale\_available = FALSE) έτσι ώστε να μην επιτρέπεται η ταυτόχρονη ενεργοποίηση του εισιτηρίου και η εισαγωγή του στην ουρά μεταπώλησης.

Ακόμη χρησιμοποιήσαμε UNIQUE για να εξασφαλίσουμε την μοναδικότητα συνδυασμών foreign keys σε junction tables και μεταξύ του festival-festival\_location για να έχουμε one-to-one relationship.

Για την αντικατάσταση του τύπου δεδομένων ENUM υλοποιήσαμε TRIGGERS για τον έλεγχο των δεδομένων που εισάγονται ή ενημερώνονται.

- Στις εμφανίσεις, να ανήκουν σε ένα από τα 3 είδη, 'warm up', 'headline', 'Special guest'
- Το προσωπικό πρέπει να ανήκει σε ένα από τα είδη, 'technicians', 'security personnel', 'support staff'
- Ο εξοπλισμός πρέπει να ανήκει σε ένα από τα 5 είδη, 'speakers', 'lights', 'microphones', 'consoles', 'special effects'
- Για τα εισιτήρια, να ανήκουν σε ένα εκ των ειδών 'general admission', 'VIP', 'backstage'
- Η μέθοδος πληρωμής πρέπει να ανήκει σε ένα από τα πεδία, 'credit card', 'debit card', 'bank deposit'

Επίσης, υλοποιήσαμε ένα FUNCTION (check\_vip) για τον έλεγχο οτι τα VIP tickets δεν υπερβαίνουν το 10% της χωρητικότητας της σκηνής. Χρησιμοποιώντας το παραπάνω FUNCTION υλοποιούμε TRIGGERS(vip\_limit\_insert/update) έτσι ώστε μετά από κάθε εισαγωγή και ενημέρωση στοιχείων στον πίνακα ticket να ελέγχει τον αριθμό των VIP.

Σχετικά με το staff έχουμε ένα PROCEDURE (CheckStaffRequirements) για τον έλεγχο της πληρότητας προσωπικού σε κάθε event και το οποίο μας επιστρέφει το υπάρχον και το ζητούμενο προσωπικό σε όσα event δεν πληρούν τις προϋποθέσεις. Επιπλέον, έχουμε TRIGGER (staff\_event\_check\_before\_delete/update) που με κάθε διαγραφή και ενημέρωση προσωπικού ελέγχει ότι πληρούνται οι απαιτούμενες συνθήκες (5% της χωρητικότητας ο αριθμός του προσωπικού ασφαλείας και 2% για το βοηθητικό προσωπικό).

Επιπλέον, υπάρχει TRIGGER (rating\_activated\_ticket\_check) για τον έλεγχο της εγκυρότητας των αξιολογήσεων. Το TRIGGER εξετάζει 2 βασικά σημεία:

- 1) Αν το εισιτήριο που αντιστοιχεί το IAN number είναι activated
- 2) Αν το εισιτήριο αυτό σχετίζεται με το ίδιο event στο οποίο ανήκει και η παράσταση που αξιολογείται.

TRIGGER (`performance_enforce_rules_before_insert/update`) επίσης έχει υλοποιηθεί για τον έλεγχο του ορίου συμμετοχής κάθε καλλιτέχνη (μέσω του FUNCTION `CheckConsecutiveYears`) και της παράλληλης εμφάνισης του σε άλλη σκηνή (μέσω του FUNCTION `CheckSimultaneousPerformance`).

Τέλος υπάρχει TRIGGER (`after_seller_insert_update_ticket`) το οποίο είναι υπεύθυνο να θέσει το status του εισιτηρίου κάθε πωλητή σε `available for resale` κατά την εισαγωγή του στο table.

Το Resale Queue υλοποιείται μέσω του procedure `ProcessAllResaleTransactions`. Καλώντας το procedure και εφόσον υπάρχουν buyers που ενδιαφέρονται για τα εισιτήρια των sellers πραγματοποιούνται όλες οι συναλλαγές με μια κλήση. Το procedure ξεκινάει εντοπίζοντας των πιο παλιο πωλητή και ψάχνει τον παλαιότερο αγοραστή που ενδιαφέρεται για το εισιτήριο. Αν δεν βρει κάποιον προχωράει στον επόμενο seller. Όταν υπάρχει match τότε δημιουργείται ένας καινούριος attendee και τα στοιχεία του αγοραστή καταχωρούνται στο table αυτό. Ο buyer και ο seller διαγράφονται από τα αντίστοιχα tables και η διαδικασία επαναλαμβάνεται. Τελικά καταγράφονται όλες οι συναλλαγές στο Ticket Transaction Table.

### A.2.2 Ευρετήρια

Κατά τον ορισμό των indices επικεντρωθήκαμε στα foreign keys και attributes που εμφανίζονται περισσότερο στα queries (Δεν ορίσαμε indices για τα primary keys καθώς δημιουργούνται αυτόματα. Ομοίως, τα foreign keys ορίζονται και αυτά αυτόματα, παρόλα αυτά παραθέτουμε ενδεικτικά τα πιο σημαντικά). Θέσαμε ένα ελάχιστο όριο των 2 queries στα οποία εμφανίζεται το attribute για να το κάνουμε index, έτσι ορίσαμε τα:

- `idx_staff_specialty` ON `staff(specialty)`
- `idx_festival_event_date` ON `festival_event(event_date)`
- `idx_subgenre_genre_id` ON `subgenre(genre_id)`
- `idx_band_artist_band_id` ON `band_artist(band_id)`
- `idx_performance_event_id` ON `performance(event_id)`
- `idx_festival_location_id` ON `festival(location_id)`
- `idx_staff_event_staff_id` ON `staff_event(staff_id)`
- `idx_artist_subgenre_artist_id` ON `artist_subgenre(artist_id)`
- `idx_festival_event_festival_year` ON `festival_event(festival_year)`

Αποφύγαμε την δημιουργία περαιτέρω indices καθώς επιβαρύνουν τις λειτουργίες εγγραφής, ενημέρωσης και διαγραφής. Επίσης, αν και αρχικά είχαμε συμπεριλάβει τα 2 παρακάτω composite indices δεν τα έχουμε στην τελική υλοποίηση καθώς χρησιμοποιήθηκαν σπάνια:

- `idx_ticket_event_purchase` ON `ticket (event_id, purchase_method);`
- `idx_perf_type_event` ON `performance (performance_type, event_id);`

## DDL και DML scripts

Για το κομμάτι του DDL υλοποιήσαμε με την σειρά τα αρχεία festival\_database.sql (schema του database με τα αντίστοιχα drops πριν από κάθε table για ευκολία κατά την εισαγωγή του στο RDBMS), procedures.sql (που περιέχει τα triggers, functions, procedures) και indices.sql (τα απαραίτητα ευρετήρια). Όσον αφορά το DML, για την δημιουργία των fake data (fake\_data.sql) χρησιμοποιήθηκε LLM που έφτιαξε το κατάλληλο python script, με χρήση της βιβλιοθήκης faker. Τέλος, υλοποιήσαμε τα queries.

## B. Queries

Τα queries βρίσκονται σε αριθμημένα αρχεία .sql με το αντίστοιχο αποτέλεσμα σε αριθμημένο αρχείο .txt

Για το query 4 χρησιμοποιούμε τον δείκτη idx\_rating\_1 ON rating(performance\_id, overall\_impression, artist\_performance). Ο πρώτος πίνακας είναι με force ο δεύτερος χωρίς τον δείκτη.

id	select_type	table	type	possible_keys	key	key_len	ref	rows	r_rows	filtered	r_filtered	Extra
1	PRIMARY	A	const	PRIMARY	PRIMARY	4	const	1	NULL	100.00	NULL	
1	PRIMARY	<derived2>	ALL	NULL	NULL	NULL	NULL	6	11.00	100.00	100.00	
1	PRIMARY	R	ref	idx_rating_1	idx_rating_1	4	API.performance_id	53	104.82	100.00	100.00	Using index
2	DERIVED	p	ref	artist_id	artist_id	5	const	2	2.00	100.00	100.00	Using index
3	UNION	ba	ref	artist_id,idx_band_artist_band_id	artist_id	4	const	1	1.00	100.00	100.00	Using index
3	UNION	p	ref	band_id	band_id	5	festival_database.ba.band_id	4	9.00	100.00	100.00	Using index
NULL UNION RESULT <union2,3>								NULL	11.00	NULL	NULL	

id	select_type	table	type	possible_keys	key	key_len	ref	rows	r_rows	filtered	r_filtered	Extra
1	PRIMARY	A	const	PRIMARY	PRIMARY	4	const	1	NULL	100.00	NULL	
1	PRIMARY	<derived2>	ALL	NULL	NULL	NULL	NULL	6	11.00	100.00	100.00	
1	PRIMARY	R	ref	performance_id	performance_id	4	API.performance_id	53	104.82	100.00	100.00	
2	DERIVED	p	ref	artist_id	artist_id	5	const	2	2.00	100.00	100.00	Using index
3	UNION	ba	ref	artist_id,idx_band_artist_band_id	artist_id	4	const	1	1.00	100.00	100.00	Using index
3	UNION	p	ref	band_id	band_id	5	festival_database.ba.band_id	4	9.00	100.00	100.00	Using index
NULL UNION RESULT <union2,3>								NULL	11.00	NULL	NULL	

Από τους παραπάνω πίνακες βλέπουμε ότι η χρήση του νέου index δεν επιταχύνει ούτε επιβραδύνει το query, πράγμα που επιβεβαιώνεται και από τα παρακάτω traces.

Ελέγχοντας τα αντίστοιχα traces χρησιμοποιώντας τις εντολές

```
SET optimizer_trace="enabled=on";
```

Και ύστερα

```
SELECT * FROM INFORMATION_SCHEMA.OPTIMIZER_TRACE;
```

Με force index:

"rows\_for\_plan": 11,

"cost\_for\_plan": 12.75,

"estimated\_join\_cardinality": 11

Ενώ αρχικά:

"rows\_for\_plan": 11,

"cost\_for\_plan": 12.75,

"estimated\_join\_cardinality": 11

Για το query 6 επιλέξαμε Force Index idx\_ticket\_attendee\_activated ON ticket (attendee\_id, activated). Το query φιλτράρει άμεσα βάσει attendee\_id = 1 και activated = TRUE οπότε συνδυάζοντας αυτά τα δύο πεδία στο index εντοπίζουμε μόνο τις εγγραφές του επισκέπτη που έχουν ενεργοποιηθεί αποφεύγοντας να σαρώσουμε όλο τον πίνακα.

id	select_type	table	type	possible_keys	key	key_len	ref	rows	r_rows	filtered	r_filtered	Extra
1	SIMPLE	T	ref	idx_ticket_attendee_activated	idx_ticket_attendee_activated	5	const,const	2	2.00	100.00	100.00	Using temporary; Using filesort
1	SIMPLE	FE	eq_ref	PRIMARY	PRIMARY	4	festival_database.T.event_id	1	1.00	100.00	100.00	Using index
1	SIMPLE	P	ref	PRIMARY,event_id	event_id	4	festival_database.T.event_id	1	1.50	100.00	100.00	Using index
1	SIMPLE	R	ref	IAN_number,performance_id	IAN_number	8	festival_database.T.IAN_number	1	1.00	100.00	66.67	Using where

id	select_type	table	type	possible_keys	key	key_len	ref	rows	r_rows	filtered	r_filtered	Extra
1	SIMPLE	T	ref	PRIMARY,event_id,attendee_id	attendee_id	4	const	2	2.00	100.00	100.00	Using where; Using temporary; Using filesort
1	SIMPLE	FE	eq_ref	PRIMARY	PRIMARY	4	festival_database.T.event_id	1	1.00	100.00	100.00	Using index
1	SIMPLE	P	ref	PRIMARY,event_id	event_id	4	festival_database.T.event_id	1	1.50	100.00	100.00	Using index
1	SIMPLE	R	ref	IAN_number,performance_id	IAN_number	8	festival_database.T.IAN_number	1	1.00	100.00	66.67	Using where

Όπως φαίνεται από τους παραπάνω πίνακες δεν παρατηρείται βελτίωση στην απόδοση με την χρήση του νέου index.

Ελέγχοντας τα αντίστοιχα traces χρησιμοποιώντας τις εντολές

```
SET optimizer_trace="enabled=on";
```

Και ύστερα

```
SELECT * FROM INFORMATION_SCHEMA.OPTIMIZER_TRACE;
```

Με force index:

"best\_join\_order": ["T", "FE", "P", "R"]

"rows\_for\_plan": 9761,

"cost\_for\_plan": 2049.2

Αρχικά:

"best\_join\_order": ["T", "FE", "P", "R"]

"rows\_for\_plan": 9761,

"cost\_for\_plan": 2049.2