ΠΑΝΕΠΙΣΤΗΜΙΟ ΠΕΙΡΑΙΩΣ

ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ

ΒΑΣΕΙΣ ΔΕΔΟΜΕΝΩΝ - ΕΡΓΑΣΙΑ ΕΤΟΥΣ 2022-2023

Κοτσοβού Πηνελόπη- Π21069

Αμπατζίογλου Γιάννης- Π21002

Κοντοπούλου Δέσποινα-Π21066

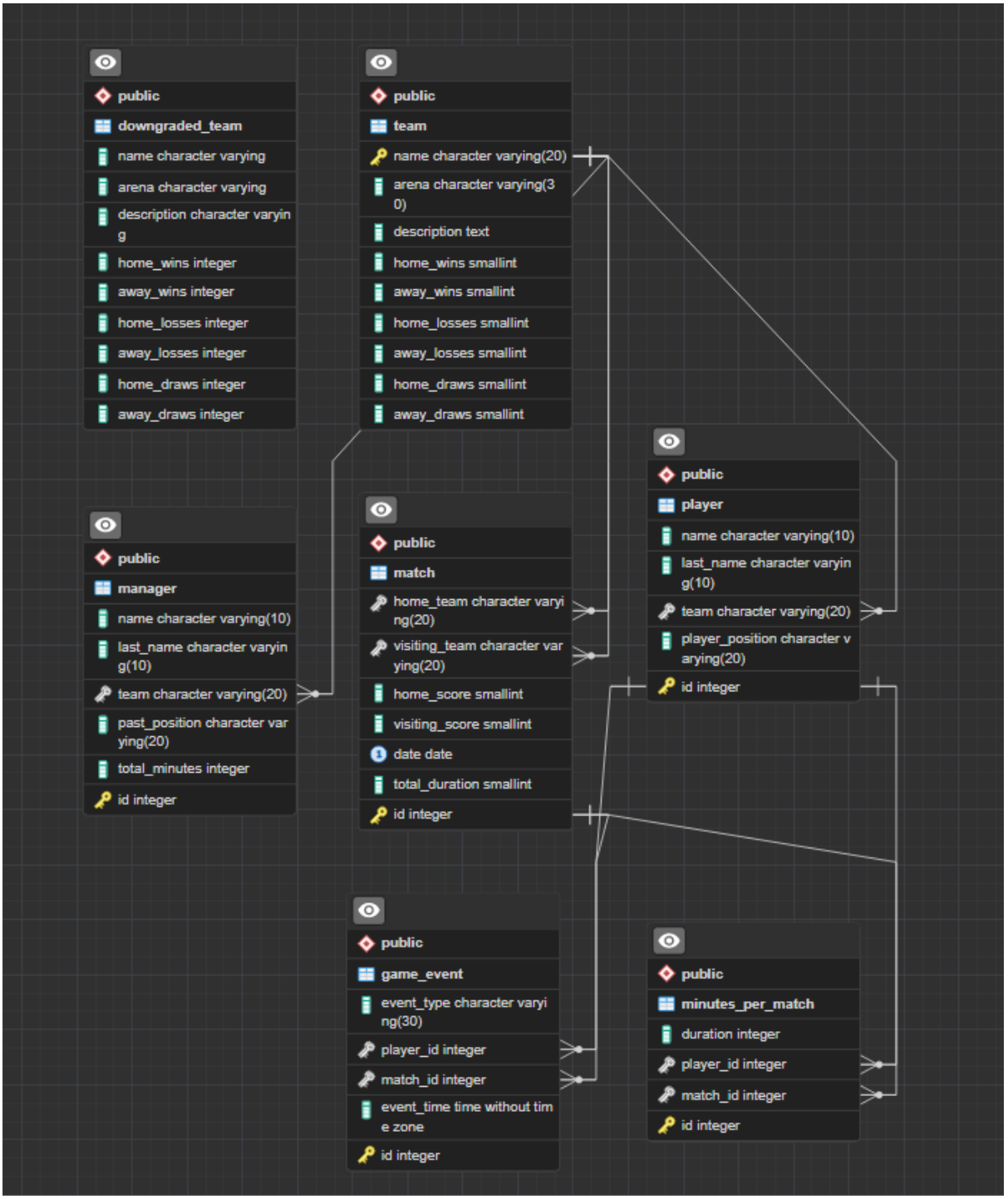
Ημερομηνία παράδοσης: 30/6/2023

1ο Ερώτημα

Εκφώνηση Εργασίας

a. Με βάση τα παραπάνω στοιχεία, σχεδιάστε το σχεσιακό σχήμα της ΒΔ, υλοποιήστε το (εντολές CREATE TABLE) στο ΣΔΒΔ PostgreSQL και φορτώστε με δεδομένα τους πίνακες. Ενδεχομένως να χρειαστεί να υλοποιήσετε επιπλέον βοηθητικούς πίνακες, σε σχέση με αυτούς οι οποίοι περιγράφονται στην εισαγωγή. Επιπλέον, καλείστε να τεκμηριώσετε τους περιορισμούς ακεραιότητας των πινάκων (και να δηλώσετε τυχόν περιορισμούς που προκύπτουν από την εκφώνηση αλλά δεν μπορέσατε να υποστηρίξετε μέσα από τους περιορισμούς ακεραιότητας των πινάκων). Το παραδοτέο του υποερωτήματος είναι το σχεσιακό σχήμα της ΒΔ, οι εντολές CREATE TABLE και τα αρχεία τα οποία θα εισάγετε στους πίνακες.

- Δημιουργήσαμε τους εξής πίνακες: team, player,manager, match, minutes_per_match, game_event και downgraded team.
- Για να ελέγχουμε ότι οι παίκτες σε κάθε ομάδα θα είναι μέχρι 11 και δεν θα μπορούμε να εισάγουμε περισσότερους δημιουργήσαμε ένα trigger με όνομα trigger_max_players το οποίο όταν εισάγεται ή ανανεώνεται κάποιος παίκτης καλεί το function check_max_players()
- Για να υπάρχει διάστημα 10 ημερών μεταξύ των αγώνων δημιουργήσαμε τα εξής: check_min_days() και trigger_min_days
- Για να επαληθεύεται οτι οι manager είναι και παλιοί παίκτες δημιουργήσαμε την function promote_to_manager(player_id INT) η οποία παίρνει σαν όρισμα το id του player


public
downgraded_team
name character varying
arena character varying
description character varying
home_wins integer
away_wins integer
home_losses integer
away_losses integer
home_draws integer
away_draws integer
public
team
name character varying(20)
arena character varying(30)
description text
home_wins smallint
away_wins smallint
home_losses smallint
away_losses smallint
home_draws smallint
away_draws smallint
public
player
name character varying(10)
last_name character varying(10)
team character varying(20)
player_position character varying(20)
id integer
public
manager
name character varying(10)
last_name character varying(10)
team character varying(20)
past_position character varying(20)
total_minutes integer
id integer
public
match
home_team character varying(20)
visiting_team character varying(20)
home_score smallint
visiting_score smallint
date date
total_duration smallint
id integer
public
game_event
event_type character varying(30)
player_id integer
match_id integer
event_time time without time zone
id integer
public
minutes_per_match
duration integer
player_id integer
match_id integer
id integer

c. Πάνω στο τελικό σχήμα της ΒΔ υλοποιήστε 2 προβολές/όψεις (views):

1. Πρόγραμμα-αγώνων. Μια προβολή που θα αφορά μια συγκεκριμένη ημερομηνία (π.χ. 30/5/2023) και θα περιλαμβάνει τις «δυναμικές» πληροφορίες των αγώνων εκείνης της ημέρας: τόπος διεξαγωγής αγώνα, χρόνος, ποιες ομάδες συμμετέχουν, ποιο το σκορ, ποιοι παίκτες από κάθε ομάδα (όνομα θέση, στο παιχνίδι, χρόνος συμμετοχής στο παιχνίδι, τις κάρτες που τυχόν χρεώθηκε, τα γκολ που έβαλε και πότε τα έβαλε).

2. Ετήσιο πρωτάθλημα αγώνων. Μια προβολή που θα αφορά μια συγκεκριμένη αγωνιστική σεζόν (π.χ. 1/9/2022 – 30/6/2023) και θα περιλαμβάνει τις «στατικές» πληροφορίες των αγώνων εκείνου του διαστήματος: τόπος διεξαγωγής αγώνα, χρόνος, ποιες ομάδες συμμετέχουν, ποιο το σκορ μεταξύ τους, ποια ομάδα είναι εντός/εκτός έδρας.

Απάντηση για c1:

```
CREATE VIEW match_schedule AS
SELECT thismatch.date AS match_date,
       thismatch.total_duration AS duration,
       home_team.arena AS arena,
       thismatch.home_team AS home_team_name,
       thismatch.visiting_team AS visiting_team_name,
       thismatch.home_score,
       thismatch.visiting_score,
       thismatchplayer.name || ' ' || thismatchplayer.last_name AS player_name,
       thismatchplayer.player_position,
       minutes.duration AS player_duration,
       CASE WHEN game_events.player_id = thismatchplayer.id THEN
game_events.event_type ELSE NULL END AS event_type,
       CASE WHEN game_events.player_id = thismatchplayer.id THEN
game_events.event_time ELSE NULL END AS event_time
FROM match thismatch
JOIN team home_team ON home_team.name = thismatch.home_team
JOIN team visiting_team ON visiting_team.name = thismatch.visiting_team
JOIN player thismatchplayer ON thismatchplayer.team = thismatch.home_team OR
thismatchplayer.team = thismatch.visiting_team
LEFT JOIN game_event game_events ON game_events.match_id = thismatch.id AND
game_events.player_id = thismatchplayer.id
JOIN minutes_per_match minutes ON minutes.match_id = thismatch.id AND
minutes.player_id = thismatchplayer.id
WHERE thismatch.date = '2021-04-08';
```

Δημιουργήσαμε ένα view με όνομα "match_schedule" που αποθηκεύεται σαν πίνακας και εκτελείται σαν query έτσι ώστε να μπορούμε να το χρησιμοποιήσουμε αργότερα. Περιλαμβάνει στις στήλες τα ακόλουθα:

match_date, duration, arena, home_team_name, visiting_team_name, home_score, visiting_score, player_name, player_position, player_duration, event_type, event_time

Εκτελείται ένα πολύπλοκο query που συνδυάζει πολλούς πίνακες. Έχουμε τα ακόλουθα join:

Μεταξύ των "match" και "team" χρησιμοποιώντας τα πεδία home_team και name αντίστοιχα ώστε να παίρνουμε τις πληροφορίες των ομάδων που συμμετέχουν στον αγώνα.

Μεταξύ των πινάκων "match" και "player" χρησιμοποιώντας τα πεδία home_team και team ή visiting_team και team αντίστοιχα ώστε να παίρνουμε τις πληροφορίες των παικτών που συμμετέχουν στον αγώνα.

Μεταξύ των "match" και "game_event" χρησιμοποιώντας τα πεδία id και match_id ή id και player_id αντίστοιχα ώστε να παίρνουμε τις πληροφορίες των συμβάντων που συμβαίνουν στον αγώνα.

Μεταξύ των "match" και "minutes_per_match" χρησιμοποιώντας τα πεδία id και match_id ή id και player_id αντίστοιχα ώστε να παίρνουμε τις πληροφορίες για τη διάρκεια που παίζει ο κάθε παίκτης στον αγώνα.

Περιορίζονται τα αποτελέσματα στην ημερομηνία '2021-04-08'.

Όταν εκτελείται το query, εμφανίζεται ο παρακάτω πίνακας με τις παραπάνω στήλες και τους αντίστοιχους περιορισμούς.

	match_date date	duration smallint	arena character varying	home_team_name character varying	visiting_team_name character varying	home_score smallint	visiting_score smallint	player_name text	player_position character varying	player_duration integer	event_type character varying	event_time time witho
1	2021-04-08	95	Agia Sofia	AEK	Panathinaikos	2	2	Χαριλαος Γιασαλιος	Center Forward	5	foul	20:25:47
2	2021-04-08	95	Agia Sofia	AEK	Panathinaikos	2	2	Πολυζώης Παπακώστ...	Center Back	61	[null]	[null]
3	2021-04-08	95	Agia Sofia	AEK	Panathinaikos	2	2	Δήμος Δελής	Left Back	86	[null]	[null]
4	2021-04-08	95	Agia Sofia	AEK	Panathinaikos	2	2	Δαμιανός Σημηριώτης	Right Back	69	[null]	[null]
5	2021-04-08	95	Agia Sofia	AEK	Panathinaikos	2	2	Σόλων Ιωαννίδης	Goal Keeper	63	[null]	[null]
6	2021-04-08	95	Agia Sofia	AEK	Panathinaikos	2	2	Αλέξιος Σαμαράς	Center Midfielder	48	[null]	[null]
7	2021-04-08	95	Agia Sofia	AEK	Panathinaikos	2	2	Πολυζώης Πολίτης	Defensive Midfielder	75	[null]	[null]
8	2021-04-08	95	Agia Sofia	AEK	Panathinaikos	2	2	Αλέξιος Βούρος	Attacking Midfielder	89	[null]	[null]
9	2021-04-08	95	Agia Sofia	AEK	Panathinaikos	2	2	Αλέξιος Δασκαλάκης	Right Wing	41	penalty kick	19:49:52
10	2021-04-08	95	Agia Sofia	AEK	Panathinaikos	2	2	Αντώνης Κορνάρος	Left Wing	9	[null]	[null]
11	2021-04-08	95	Agia Sofia	AEK	Panathinaikos	2	2	Ρομπέρτο Γιαλαμάς	Center Forward	84	[null]	[null]
12	2021-04-08	95	Agia Sofia	AEK	Panathinaikos	2	2	Σοφοκλής Φανουράκης	Center Back	56	[null]	[null]
13	2021-04-08	95	Agia Sofia	AEK	Panathinaikos	2	2	Θωμάς Τρικούπης	Left Back	81	[null]	[null]
14	2021-04-08	95	Agia Sofia	AEK	Panathinaikos	2	2	Δημήτρης Ιακωβίδης	Right Back	65	[null]	[null]
15	2021-04-08	95	Agia Sofia	AEK	Panathinaikos	2	2	Ορφέας Τσουκαλάς	Goal Keeper	65	[null]	[null]
16	2021-04-08	95	Agia Sofia	AEK	Panathinaikos	2	2	Αλέξιος Δημαράς	Center Midfielder	66	[null]	[null]
17	2021-04-08	95	Agia Sofia	AEK	Panathinaikos	2	2	Ερμής Πάριος	Defensive Midfielder	54	[null]	[null]
18	2021-04-08	95	Agia Sofia	AEK	Panathinaikos	2	2	Ρομπέρτο Βούλγαρης	Attacking Midfielder	77	[null]	[null]
19	2021-04-08	95	Agia Sofia	AEK	Panathinaikos	2	2	Αλέξιος Πέτσας	Right Wing	10	[null]	[null]
20	2021-04-08	95	Agia Sofia	AEK	Panathinaikos	2	2	Ρήγας Πάριος	Left Wing	41	[null]	[null]
21	2021-04-08	95	Agia Sofia	AEK	Panathinaikos	2	2	Θωμάς Ευθυμιάδης	Center Forward	33	[null]	[null]
22	2021-04-08	95	Agia Sofia	AEK	Panathinaikos	2	2	Αλέξανδρος Δυδρόου	Center Back	37	[null]	[null]

Total rows: 22 of 22 Query complete 00:00:00.389 Ln 1, Col 24

Απάντηση για c2:

```
CREATE VIEW league_matches AS
SELECT thismatch.total_duration AS duration,
       home_team.arena AS arena,
       thismatch.home_team AS home_team_name,
       thismatch.visiting_team AS visiting_team_name,
       thismatch.home_score,
       thismatch.visiting_score
FROM match thismatch
JOIN team home_team ON home_team.name = thismatch.home_team
WHERE thismatch.date>='2023-01-01' AND thismatch.date<='2023-12-30';
```

Δημιουργήσαμε ένα view με όνομα "league_matches" που αποθηκεύεται σαν πίνακας και εκτελείται σαν query έτσι ώστε να μπορούμε να το χρησιμοποιήσουμε αργότερα. Περιλαμβάνει στις στήλες τα ακόλουθα:

duration, arena, home_team_name, visiting_team_name, home_score, visiting_score.

Έχουμε ένα join μεταξύ του "match" και του "team" το οποίο ελέγχει ότι το όνομα της γηπεδούχου ομάδας είναι ίδιο με το όνομα της ομάδας στον πίνακα "team"

Η συνθήκη στο WHERE περιορίζει τα αποτελέσματα μόνο στους αγώνες που έχουν ημερομηνία μεταξύ '2023-04-01' και '2023-04-30'.

Όταν εκτελείται το query, εμφανίζεται ο παρακάτω πίνακας με τις παραπάνω στήλες και τους περιορισμούς.

Data Output | Messages | Notifications

	duration smallint	arena character varying	home_team_name character varying	visiting_team_name character varying	home_score smallint	visiting_score smallint
1	90	Agia Sofia	AEK	Volos	0	3
2	95	Apostollos Nikolaidis	Panathinaikos	PAOK	2	1
3	90	Apostollos Nikolaidis	Panathinaikos	AEK	1	0
4	90	Toubas	PAOK	Olympiacos	2	2
5	96	Toubas	PAOK	AEK	3	3
6	89	Georgios Karaiskakis	Olympiacos	Panathinaikos	2	0
7	89	Kleanthis Bikelidis	Aris	Olympiacos	4	4

2ο Ερώτημα

Εκφώνηση Εργασίας

a) Ποιος είναι προπονητής μιας συγκεκριμένης ομάδας σε συγκεκριμένο αγώνα;

```
SELECT manager.name , manager.last_name
FROM manager
JOIN team ON team.name=manager.team
JOIN match ON (team.name=match.home_team )--ή θα μπορούσε να
είναι : team.name=match.visiting_team
WHERE match.id=3
```

Κάνουμε join τον πίνακα **manager** με :

-> τον πίνακα **team** στο σημείο όπου το team.name έχει ίδια τιμή με την team του manager

->τον πίνακα **match** στο σημείο όπου το team.name έχει την ίδια τιμή είτε με την home_team είτε με την visiting_team του πίνακα match

και εμφανίζουμε το name και το last_name του προπονητή από τον πίνακα manager,

εκεί όπου το id του πίνακα match είναι ίσο με έναν από το διάστημα αριθμών [1-30] (δηλαδή κάθε φορά επιλέγουμε σε ποιον συγκεκριμένο αγώνα από τους 30 που υπάρχουν θέλουμε να μάθουμε τα ονοματεπώνυμα των προπονητών των αντίστοιχων ομάδων).

Query Query History

```
SELECT manager.name , manager.last_name
FROM manager
JOIN team ON team.name=manager.team
JOIN match ON (team.name=match.home_team )--ή θα μπορούσε να είναι : team.name=match.visiting_team
WHERE match.id=3
```

Data Output Messages Notifications

	name character varying	last_name character varying
1	Αθανάσιος	Κεδίκογλου

b) Τα γκολ, πέναλτι που έγιναν σε συγκεκριμένο αγώνα, ποια χρονική στιγμή και από ποιόν παίκτη.

```
SELECT game_event.event_type, game_event.event_time,
player.name AS player_name
FROM game_event
JOIN player ON game_event.player_id = player.id
JOIN match on game_event.match_id = match.id
WHERE match.id = '5' AND (game_event.event_type = 'goal'
OR game_event.event_type = 'penalty kick');
```

Λογική:

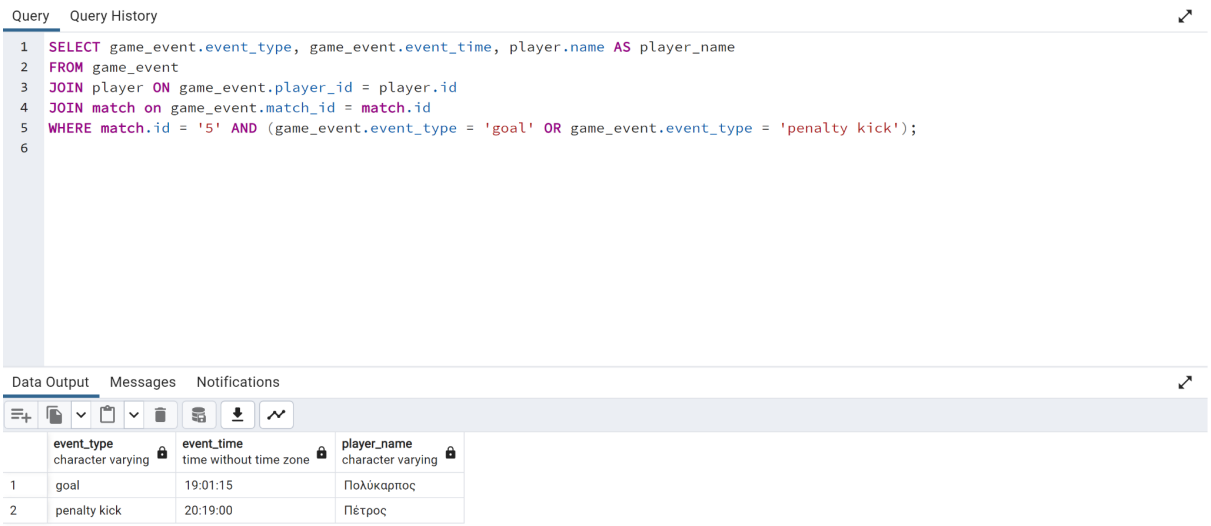
Από τον πίνακα game_event κάνοντας τον join με:

- τον πίνακα player στα σημεία που το player_id του πίνακα game_event έχει ίδια τιμή με το id του πίνακα player

- τον πίνακα match στα σημεία που το match_id του πίνακα game_event έχει ίδια τιμή με το id του πίνακα match,

εμφανίζουμε game_event.event_type (goal / penalty kick), game_event.event_time, player.name

όπου match.id είναι 5 και game_event.event_type είναι goal ή penalty kick.



c) Την αγωνιστική εικόνα ενός συγκεκριμένου παίκτη για μια αγωνιστική σεζόν: γκολ, πέναλτι, κάρτες, λεπτά αγώνα, θέση που έπαιξε.

```
SELECT p.id as player_id,p.name as player_first_name,
p.last_name AS player_last_name, p.team,p.player_position,
m.id AS match_id,min.duration AS
minutes_per_player_per_match,
CASE WHEN game_events.match_id =m.id and
game_events.player_id=p.id THEN game_events.event_type ELSE
NULL END AS event_type
FROM player p
JOIN match m on p.team=m.home_team or p.team=m.visiting_team
JOIN minutes_per_match min on min.player_id=p.id and
min.match_id=m.id
LEFT JOIN game_event game_events on
game_events.player_id=p.id
WHERE m.id IN (SELECT match.id FROM match WHERE EXTRACT(YEAR
FROM match.date) = 2023) AND p.id = 3 ORDER BY m.id;
```

Κάνω join τον πίνακα **player AS p** με :

-> τον πίνακα **match AS m**στο σημείο όπου το **p.team** έχει ίδια τιμή με το **m.home_team** ή **p.team** έχει ίδια τιμή με το **m.visiting_team**

->τον πίνακα **minutes_per_match AS min** στο σημείο όπου το **min.player_id** έχει ίδια τιμή με το **p.id** και το **min.match_id** έχει ίδια τιμή με το **m.id**

Κάνω left join τον πίνακα **player AS p** με :

->τον πίνακα **game_event AS game_events** στο σημείο όπου το **game_events.player_id** έχει ίδια τιμή με το **p.id**

και εμφανίζω το **p.id as player_id,** το **p.name as player_first_name,** το **p.last_name AS player_last_name,** το **p.team,p.player_position,** το **m.id AS match_id,** το **min.duration AS minutes_per_player_per_match,**

και κάνω **CASE WHEN game_events.match_id =m.id and game_events.player_id=p.id THEN game_events.event_type ELSE NULL END AS event_type** δηλαδή προβάλλεται το event_type από τον πίνακα game_events όταν ικανοποιείται η συνθήκη **game_events.match_id =m.id** και **game_events.player_id=p.id.**Αν η συνθήκη αυτή είναι αληθής τότε εμφανίζεται η τιμή του event_type αλλιώς η τιμή NULL.

Query | Query History

```
select p.id as player_id,p.name as player_first_name, p.last_name as player_last_name, p.team,p.player_position, m.id as match_id,min
    CASE WHEN game_events.match_id =m.id and game_events.player_id=p.id THEN game_events.event_type ELSE NULL END AS event_type
    from player p
    join match m on p.team=m.home_team or p.team=m.visiting_team
    join minutes_per_match min on min.player_id=p.id and min.match_id=m.id
    left join game_event game_events on game_events.player_id=p.id
    where m.id IN (
    SELECT match.id
    FROM match
    WHERE EXTRACT(YEAR FROM match.date) = 2023) and p.id = 3 order by m.id ;
```

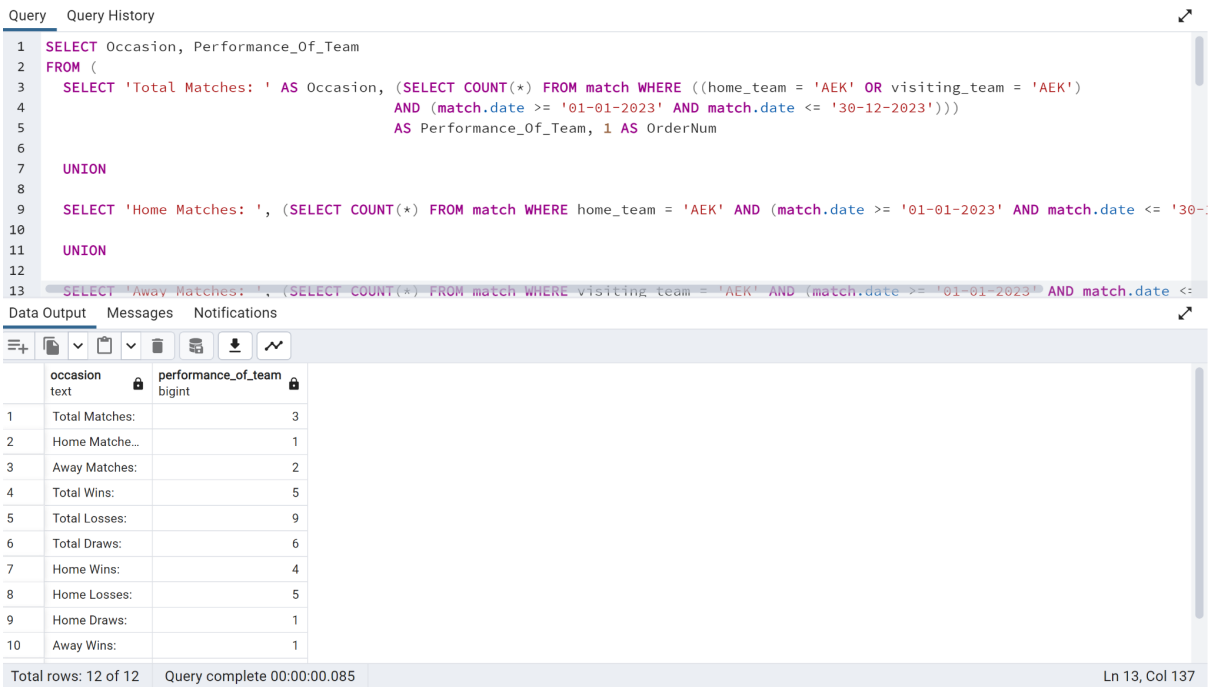
Data Output | Messages | Notifications

	player_id integer	player_first_name character varying	player_last_name character varying	team character varying	player_position character varying	match_id integer	minutes_per_player_per_match integer	event_type character varying
1	3	Δήμος	Δελής	AEK	Left Back	5	20	[null]
2	3	Δήμος	Δελής	AEK	Left Back	6	79	[null]
3	3	Δήμος	Δελής	AEK	Left Back	11	66	goal

d) Την αγωνιστική εικόνα μιας συγκεκριμένης ομάδας για μια αγωνιστική σεζόν: σε πόσους αγώνες συμμετείχε, σε πόσους ήταν γηπεδούχος και σε πόσους φιλοξενούμενη, πόσες ήττες /νίκες/ ισοπαλίες, πόσες φορές νίκησε/ έχασε/ έφερε ισοπαλία εντός/ εκτός έδρας.

```sql
SELECT Occasion, Performance_Of_Team
FROM (
SELECT 'Total Matches: ' AS Occasion, (SELECT COUNT(*) FROM match WHERE
((home_team = 'AEK' OR visiting_team = 'AEK') AND (match.date >=
'01-01-2023' AND match.date <= '12-30-2023'))) AS Performance_Of_Team, 11 AS
Order_Num
UNION
SELECT 'Home Matches: ', (SELECT COUNT(*) FROM match WHERE home_team = 'AEK'
AND (match.date >= '01-01-2023' AND match.date <= '12-30-2023')), 2
UNION
SELECT 'Away Matches: ', (SELECT COUNT(*) FROM match WHERE visiting_team =
'AEK' AND (match.date >= '01-01-2023' AND match.date <= '12-30-2023')), 3
UNION
SELECT 'Total Wins: ', (SELECT SUM(home_wins + away_wins) FROM team WHERE
name = 'AEK'), 4
UNION
SELECT 'Total Losses: ', (SELECT SUM(home_losses + away_losses) FROM team
WHERE name = 'AEK'), 5
UNION
SELECT 'Total Draws: ', (SELECT SUM(home_draws + away_draws) FROM team WHERE
name = 'AEK'), 6
UNION
SELECT 'Home Wins: ', (SELECT home_wins FROM team WHERE name = 'AEK'), 7
UNION
SELECT 'Home Losses: ', (SELECT home_losses FROM team WHERE name = 'AEK'), 8
UNION
SELECT 'Home Draws: ', (SELECT home_draws FROM team WHERE name = 'AEK'), 9
UNION
SELECT 'Away Wins: ', (SELECT away_wins FROM team WHERE name = 'AEK'), 10
UNION
SELECT 'Away Losses: ', (SELECT away_losses FROM team WHERE name = 'AEK'),
11
UNION
SELECT 'Away Draws: ', (SELECT away_draws FROM team WHERE name = 'AEK'), 12)
AS sub ORDER BY Order_Num;
```

Λογική: Θέλαμε να εμφανίσουμε τα αποτελέσματα της ομάδας σε δύο στήλες Occasion και Performance_Of_Team. Η Occasion περιγράφει το αποτέλεσμα και η Performance_Of_Team εμφανίζει τον αριθμό του αποτελέσματος. Οπότε ξεκινήσαμε δημιουργώντας το εσωτερικό query το οποίο με την χρήση του UNION διευκολύνει την εμφάνιση των αποτελεσμάτων αφού προσθέτει κάθε φορά μία σειρά. Κάθε καινούργια σειρά που εισάγεται έχει σαν παράμετρο ένα String το οποίο ανήκει στη στήλη του Occasion, ένα εσωτερικό Query το οποίο υπολογίζει με τους κατάλληλους περιορισμούς το αποτέλεσμα με όνομα Performance_Of_Team και έναν αριθμό ο οποίος θα αυξάνεται σειριακά με όνομα μεταβλητής Order_Num. Το UNION επειδή ακριβώς προσθέτει μία έξτρα σειρά στις ήδη υπάρχουσες στήλες, μας επιτρέπει να μην γράφουμε σε κάθε SELECT το όνομα της κάθε στήλης.



Query Query History

```
1  SELECT Occasion, Performance_Of_Team
2  FROM (
3    SELECT 'Total Matches: ' AS Occasion, (SELECT COUNT(*) FROM match WHERE ((home_team = 'AEK' OR visiting_team = 'AEK')
4                                        AND (match.date >= '01-01-2023' AND match.date <= '30-12-2023')))
5                                        AS Performance_Of_Team, 1 AS OrderNum
6
7    UNION
8
9    SELECT 'Home Matches: ', (SELECT COUNT(*) FROM match WHERE home_team = 'AEK' AND (match.date >= '01-01-2023' AND match.date <= '30-
10
11   UNION
12
13   SELECT 'Away Matches: ', (SELECT COUNT(*) FROM match WHERE visiting_team = 'AEK' AND (match.date >= '01-01-2023' AND match.date <=
```

Data Output Messages Notifications

	occasion text	performance_of_team bigint
1	Total Matches:	3
2	Home Matche...	1
3	Away Matches:	2
4	Total Wins:	5
5	Total Losses:	9
6	Total Draws:	6
7	Home Wins:	4
8	Home Losses:	5
9	Home Draws:	1
10	Away Wins:	1

Total rows: 12 of 12 Query complete 00:00:00.085 Ln 13, Col 137

3ο Ερώτημα

Εκφώνηση Εργασίας

a. Φτιάξτε έναν trigger ο οποίος κρατά/γεμίζει ένα πίνακα-ιστορικό. Όταν διαγράφονται με επιτυχία γραμμές από τον πίνακα ομάδες (π.χ. διαγράφονται όλες οι ομάδες οι οποίες δεν πέτυχαν καμία νίκη μέσα στο έτος) τότε οι διαγραμμένες γραμμές εισάγονται αυτόματα στον πίνακα ομάδες-υποβιβασμός-κατηγορίας.

```
CREATE TABLE downgraded_team (
    name VARCHAR,
    arena VARCHAR,
    description VARCHAR,
    home_wins INTEGER,
    away_wins INTEGER,
    home_losses INTEGER,
    away_losses INTEGER,
    home_draws INTEGER,
    away_draws INTEGER
);

CREATE OR REPLACE FUNCTION downgrade_team(team_to_downgrade
VARCHAR)
RETURNS VOID AS $$
BEGIN
    -- Delete rows from tables that reference the team to downgrade
     DELETE FROM game_event WHERE player_id IN (select player.id
from player where player.team=team_to_downgrade);
     DELETE FROM game_event WHERE match_id IN (select match.id from
match where (match.home_team=team_to_downgrade or
match.visiting_team=team_to_downgrade));
     DELETE FROM minutes_per_match WHERE player_id IN (select
player.id from player where player.team=team_to_downgrade);
     DELETE FROM minutes_per_match WHERE match_id IN (select
match.id from match where (match.home_team=team_to_downgrade or
match.visiting_team=team_to_downgrade));
     DELETE FROM match where (home_team=team_to_downgrade or
visiting_team=team_to_downgrade);
     DELETE FROM manager WHERE team = team_to_downgrade;
```

```
    DELETE FROM player WHERE team = team_to_downgrade;

    -- Insert team into downgraded_team
    INSERT INTO downgraded_team (name, arena, description,
home_wins, away_wins, home_losses, away_losses, home_draws,
away_draws)
    SELECT name, arena, description, home_wins, away_wins,
home_losses, away_losses, home_draws, away_draws
    FROM team
    WHERE name = team_to_downgrade;

END;
$$ LANGUAGE plpgsql;
CREATE OR REPLACE FUNCTION downgrade_team_trigger_function()
RETURNS TRIGGER AS $$
BEGIN
    PERFORM downgrade_team(OLD.name);
    RETURN OLD;
END;
$$ LANGUAGE plpgsql;

CREATE TRIGGER downgrade_team_trigger
BEFORE DELETE ON team
FOR EACH ROW
EXECUTE FUNCTION downgrade_team_trigger_function();
```

Δημιουργούμε την function downgrade_team, η οποία παίρνει ως όρισμα το όνομα της ομάδας προς υποβιβασμό. Κάνω τις απαραίτητες διαγραφές στους πίνακες που χρησιμοποιούν την ομάδα ως foreign key ή καποιο απο τα αντικείμενα που διέγραψα λογω της σχέσης foreign key. Επειτα εισάγω τα δεδομένα στον πίνακα downgraded_team. Φτιάχνουμε άλλο ένα function, το downgrade_team_trigger_function, το οποίο θα καλείται απο το trigger. Αυτο εκτελεί την μέθοδο downgrade team για το όνομα της ομάδας που διαγράφουμε. Τέλος, φτιάχνουμε ενα trigger, το downgrade_team_trigger, το οποίο καλεί την προηγούμενη συνάρτηση πριν την διαγραφή μιας πλειάδας στον πίνακα team.

b. Βρείτε για κάθε παίκτη ομαδοποιημένα ανά χρονικά διαστήματα και ανά ομάδα και ανά αγώνα τα: γκολ, πέναλτι, κάρτες, λεπτά αγώνα, θέση που έπαιξε. Χρησιμοποιείστε cursors ώστε να εμφανίσετε τις γραμμές σε ομάδες των 10.
Δεν απαντήθηκε

4ο Ερώτημα

Εκφώνηση Εργασίας

Υλοποιήστε προγραμματιστικά έναν client σε οποιαδήποτε γλώσσα προγραμματισμού γνωρίζετε (π.χ. Python, Java, C) χρησιμοποιώντας την κατάλληλη βιβλιοθήκη σύνδεσης με την PostgreSQL (π.χ. psycopg2, JDBC, ODBC). Ο client θα συνδέεται στο ΣΔΒΔ της PostgreSQL, θα εκτελεί τα queries του Ερωτήματος 2, και θα εμφανίζει τα αποτελέσματα στον χρήστη (είτε σε terminal είτε με γραφικά).

```
import psycopg2

# Connecting to our db
conn = psycopg2.connect(
    host="localhost",
    database="Final",
    user="postgres",
    password="root"
)

cur = conn.cursor()

# Function
def execute_query(query):
     # Setting the datestyle to 'ISO, MDY' for queries that use date
    cur.execute("SET datestyle = 'ISO, MDY'")
    cur.execute(query)
    results = cur.fetchall()
    for row in results:
        print(row)

while True:
```

```
    # Ask the user for input
    print("Select a query to run:")
    print("1. Query 2α")
    print("2. Query 2β")
    print("3. Query 2γ")
    print("4. Query 2δ")
    print("0. Exit")

    user_choice = input("Enter the number of the query (or 0 to exit):
")

    if user_choice == "0":
        # Exit the program
        break
    elif user_choice == "1":
        query = "SELECT manager.name, manager.last_name FROM manager
JOIN team ON team.name=manager.team JOIN match ON
(team.name=match.home_team) WHERE match.id=3"
    elif user_choice == "2":
        query = "SELECT game_event.event_type, game_event.event_time,
player.name AS player_name FROM game_event JOIN player ON
game_event.player_id = player.id JOIN match ON game_event.match_id =
match.id WHERE match.id = '5' AND (game_event.event_type = 'goal' OR
game_event.event_type = 'penalty kick')"
    elif user_choice == "3":
        query = "SELECT p.id AS player_id, p.name AS player_first_name,
p.last_name AS player_last_name, p.team, p.player_position, m.id AS
match_id, min.duration AS minutes_per_player_per_match, CASE WHEN
game_events.match_id = m.id AND game_events.player_id = p.id THEN
game_events.event_type ELSE NULL END AS event_type FROM player p JOIN
match m ON p.team = m.home_team OR p.team = m.visiting_team JOIN
minutes_per_match min ON min.player_id = p.id AND min.match_id = m.id
LEFT JOIN game_event game_events ON game_events.player_id = p.id WHERE
m.id IN (SELECT match.id FROM match WHERE EXTRACT(YEAR FROM match.date)
= 2023) AND p.id = 3 ORDER BY m.id"
    elif user_choice == "4":
        query = "SELECT Occasion, Performance_Of_Team FROM (SELECT
'Total Matches: ' AS Occasion, (SELECT COUNT(*) FROM match WHERE
((home_team = 'AEK' OR visiting_team = 'AEK') AND (match.date >=
'2023-01-01' AND match.date <= '2023-12-30'))) AS Performance_Of_Team,
1 AS Order_Num UNION SELECT 'Home Matches: ', (SELECT COUNT(*) FROM
match WHERE home_team = 'AEK' AND (match.date >= '01-01-2023' AND
```

```
match.date <= '12-30-2023')), 2 UNION SELECT 'Away Matches: ', (SELECT
COUNT(*) FROM match WHERE visiting_team = 'AEK' AND (match.date >=
'01-01-2023' AND match.date <= '12-30-2023')), 3 UNION SELECT 'Total
Wins: ', (SELECT SUM(home_wins + away_wins) FROM team WHERE name =
'AEK'), 4 UNION SELECT 'Total Losses: ', (SELECT SUM(home_losses +
away_losses) FROM team WHERE name = 'AEK'), 5 UNION SELECT 'Total
Draws: ', (SELECT SUM(home_draws + away_draws) FROM team WHERE name =
'AEK'), 6 UNION SELECT 'Home Wins: ', (SELECT home_wins FROM team WHERE
name = 'AEK'), 7 UNION SELECT 'Home Losses: ', (SELECT home_losses FROM
team WHERE name = 'AEK'), 8 UNION SELECT 'Home Draws: ', (SELECT
home_draws FROM team WHERE name = 'AEK'), 9 UNION SELECT 'Away Wins: ',
(SELECT away_wins FROM team WHERE name = 'AEK'), 10 UNION SELECT 'Away
Losses: ', (SELECT away_losses FROM team WHERE name = 'AEK'), 11 UNION
SELECT 'Away Draws: ', (SELECT away_draws FROM team WHERE name =
'AEK'), 12) AS sub ORDER BY Order_Num"
    else:
        print("Invalid choice. Please try again.")
        continue

    # Execute the query
    execute_query(query)

# Close the cursor and connection
cur.close()
conn.close()
```

Κάνουμε import την βιβλιοθήκη psycopg2 και συνδεόμαστε με την βάση χρησιμοποιώντας τα στοιχεία της. Δημιουργούμε μία συνάρτηση execute_query που λαμβάνει ως όρισμα το αντικείμενο query, η οποία αφού θεσει το datestyle στην κατάλληλη μορφη για τα queries μας,κανει execute το query και εκτυπώνει τα αποτελέσματα. Εκτός της συνάρτησης, μεσα σε ενα while loop ζητάω απο τον χρήστη να επιλέξει ένα απο τα διαθέσιμα queries και το εκτελώ. Τέλος μόλις ο χρήστης κάνει exit κλείνω τον κέρσορα και την σύνδεση με την βαση.