

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
ІМЕНІ ІГОРЯ СІКОРСЬКОГО»
ФАКУЛЬТЕТ ІНФОРМАТИКИ ТА ОБЧИСЛЮВАЛЬНОЇ ТЕХНІКИ
(повна назва інституту/факультету)

КАФЕДРА інформатики та програмної інженерії
(повна назва кафедри)

КУРСОВА РОБОТА

з дисципліни «Бази даних»
(назва дисципліни)

на тему:

База даних для підтримки діяльності федерації футболу країни

Студента (ки) 2 курсу ІІІ-13 групи
спеціальності 121 «Інженерія програмного
забезпечення»

Ал Хадама Мурата Резговича

(прізвище та ініціали)

Керівник: Ліщук Олександр Васильович

(посада, вчене звання, науковий ступінь, прізвище та ініціали)

Національна шкала _____

Кількість балів: _____ Оцінка ECTS _____

Члени комісії

(підпис)

(вчене звання, науковий ступінь, прізвище та ініціали)

(підпис)

(вчене звання, науковий ступінь, прізвище та ініціали)

(підпис)

(вчене звання, науковий ступінь, прізвище та ініціали)

Київ – 2022 рік

**Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського»**

Факультет Інформатики та обчислювальної техніки
(повна назва)

Кафедра Інформатики та програмної інженерії
(повна назва)

Дисципліна Бази даних

Курс 2 Група ІІІ-13 Семестр 3

**З А В Д А Н Н Я
НА КУРСОВУ РОБОТУ СТУДЕНТУ**

Ал Хадаму Мурату Резговичу

(прізвище, ім'я, по батькові)

1. Тема роботи: База даних для підтримки діяльності федерації футболу країни

керівник роботи: Ліщук Олександр Васильович

(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

2. Строк подання студентом роботи

3. Вихідні дані до роботи

завдання на розробку бази даних для перефразувати те, що в темі

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити)

1) Аналіз предметного середовища

2) Побудова ER-моделі

3) Побудова реляційної схеми з ER-моделі

4) Створення бази даних, у форматі обраної системи управління базою даних

5) Створення користувачів бази даних

6) Імпорт даних з використанням засобів СУБД в створену базу даних

7) Створення мовою SQL запитів

8) Оптимізація роботи запитів

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень)

6. Дата видачі завдання: 08.11.2022

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів виконання курсового проекту	Строк виконання етапів проекту	Примітка
1	Аналіз предметного середовища	29.12.22	
2	Побудова ER-моделі	29.12.22	
3	Побудова реляційної схеми з ER-моделі	30.12.22	
4	Створення бази даних, у форматі обраної системи управління базою даних	31.12.22	
5	Створення користувачів бази даних	01.01.23	
6	Імпорт даних з використанням засобів СУБД в створену базу даних	02.01.23	
7	Створення мовою SQL запитів	05.01.23	
8	Оптимізація роботи запитів	06.01.23	
9	Оформлення пояснювальної записки	07.01.23	
10	Захист курсової роботи	11.01.23	

Студент

(підпис)

Ал Хадам М. Р.
(прізвище та ініціали)

Керівник роботи

(підпис)

Ліщук О. В.
(прізвище та ініціали)

ЗМІСТ

ВСТУП.....	5
1 Опис предметного середовища	6
2 Постановка задачі	7
3 ER-діаграма	8
3.1 Бізнес-правила	8
3.2 Вибір сутностей	8
3.3 Набори атрибутів сутностей	9
4 Реляційна модель бази даних	12
4.1 Побудова необхідних відношень, визначення первинних та зовнішніх ключів	12
5 Реалізація бази даних.....	13
5.1 Створення бази даних.....	13
5.2 Імпортування даних.....	17
6 Створення користувачів бази даних	18
6.1 Рефері	18
6.2 Вболівальник.....	18
6.3 Менеджер команди.....	18
7 SQL запити	20
7.1 Створення тригерів на таблиці	20
7.2 Створення процедур	25
7.3 Створення функцій	30
7.4 Створення представлень	33
7.5 Створення різних запитів	36
7.6 Створення індексів	49

Висновок	51
Перелік посилань.....	52

ВСТУП

У сучасному світі бази даних відіграють вирішальну роль у зберіганні, організації та управлінні величезними обсягами даних. Вони є важливим компонентом сучасних обчислювальних систем, які використовуються в широкому діапазоні галузей і застосувань.

Бази даних дозволяють організаціям ефективно зберігати та отримувати великі обсяги структурованих даних, таких як інформація про клієнтів, записи про продажі та дані про запаси. Вони також надають інструменти для організації, обробки та аналізу даних, що дає змогу отримувати цінну інформацію та приймати обґрунтовані рішення.

Крім того, бази даних є центральним сховищем даних, гарантуючи їх послідовність, точність і актуальність. Вони також підтримують заходи безпеки, такі як автентифікація користувачів і шифрування даних, щоб захистити конфіденційну інформацію від несанкціонованого доступу.

Таким чином, бази даних є невід'ємною частиною сучасних обчислювальних систем і мають важливе значення для підприємств, урядів та інших організацій для ефективного керування та використання даних. У цій курсовій роботі ми вивчимо основи баз даних, включаючи їх дизайн, структуру та функціональність, і навчимося ефективно сховище даних для зберігання, отримання та аналізу даних на прикладі бази даних для підтримки діяльності федерації футболу країни.

1 ОПИС ПРЕДМЕТНОГО СЕРЕДОВИЩА

Предметним середовищем для бази даних для підтримки футбольної федерації країни є світ футболу на національному рівні. Ця база даних зберігатиме та керуватиме даними, пов'язаними з футбольними командами, гравцями, матчами, суддями, лігами та іншими пов'язаними футбольними сутностями.

База даних для підтримки футбольної федерації країни забезпечить комплексне та централізоване сховище даних, пов'язаних з футболом, що дозволить організаціям та фанатам певних футбольних чемпіонатів ефективно зберігати, керувати та аналізувати дані про команди, гравців, матчі та інші пов'язані сутності даної бази даних.

2 ПОСТАНОВКА ЗАДАЧІ

Метою даної роботи є розробка бази даних для федерації футболу певної країни. Тобто організація даних таким чином, щоб робота футбольної федерації виконувалась максимально ефективно. Отже основні задачі:

Вболівальник повинен мати змогу:

- переглянути всі зіграні матчи та побачити їх результати;
- переглянути всі контракти погоджені між командою і гравцем та його дані.

Рефері повинен мати змогу:

- переглядати та вносити зміни до кількості жовтих, червоних карток певного гравця;
- переглядати, змінювати та видаляти відомості про футбольні матчі.

Менеджер футбольної команди повинен мати змогу:

- має можливість виконувати усі маніпуляції з таблицями, які мають відношення до його команди.

3 ER-ДІАГРАМА

3.1 Бізнес-правила

1. Гравець не може змінити собі номер, якщо має дійсний контракт з командою, та в межах цієї команди цей номер вже зайнятий.
2. Кожний спонсор може одночасно працювати з максимум двома командами за правилами федерації.
3. Гравець повинен мати лише 1 дійсний контракт з командою.
4. Гравець не може покинути базу асоціації, якщо має дійсний контракт з будь-якою командою.
5. Тренерський штаб команди може складатися з одного або декількох тренерів.

Після аналізу було виділено такі сутності та зв'язки між ними:

3.2 Вибір сутностей

- Players
- Player_Cards
- Contracts
- Teams
- Coaches
- Sponsors
- Matches
- Leagues
- Referees
- Stadiums

3.3 Набори атрибутів сутностей

Таблиця 3.1 – Сутності та їхні атрибути

Сутність	Атрибути
Players	id first_name last_name play_number
Player_Cards	id player position height weight yellow_cards red_cards birth_date nationality
Contracts	id team player start_date end_date salary
Teams	id sponsor team_name founded
Coaches	id team first_name last_name

Продовження таблиці 3.1

Сутність	Атрибути
Coaches (продовження)	experience nationality
Sponsors	id sponsor_name type
Matches	id league stadium referee home_team away_team home_score away_score match_date
Leagues	id league_name
Referees	id first_name last_name qualification
Stadiums	id stadium_name city capacity

Сутність Players буде пов'язана **один до одного** із сутністю Player_Cards, адже один гравець має відповідну лише одну картку гравця.

Сутність Players буде пов'язана **один до багатьох** з сутністю Contracts, адже один гравець може мати різні контракти в різний час.

Сутність Sponsors пов'язана **один до багатьох** з сутністю Teams, тому що один спонсор може співпрацювати одночасно з декількома командами.

Сутність Teams пов'язана **один до багатьох** з сутністю Coaches, тому що одна команда може мати багато тренерів.

Сутність Teams пов'язана **один до багатьох** з сутністю Contracts, тому що одна команда має багато контрактів з різними гравцями.

Сутність Teams пов'язана **один до багатьох** з сутністю Matches, тому що одна команда може мати багато матчів як команда-господар або команда-гість.

Сутність Stadiums пов'язана **один до багатьох** з сутністю Matches, тому що один стадіон може прийняти багато матчів.

Сутність Referees пов'язана **один до багатьох** з сутністю Matches, тому що один рефері може обслуговувати багато матчів.

Сутність League пов'язана **один до багатьох** з сутністю Matches, тому що в рамках однієї ліги може бути зіграно багато матчів.

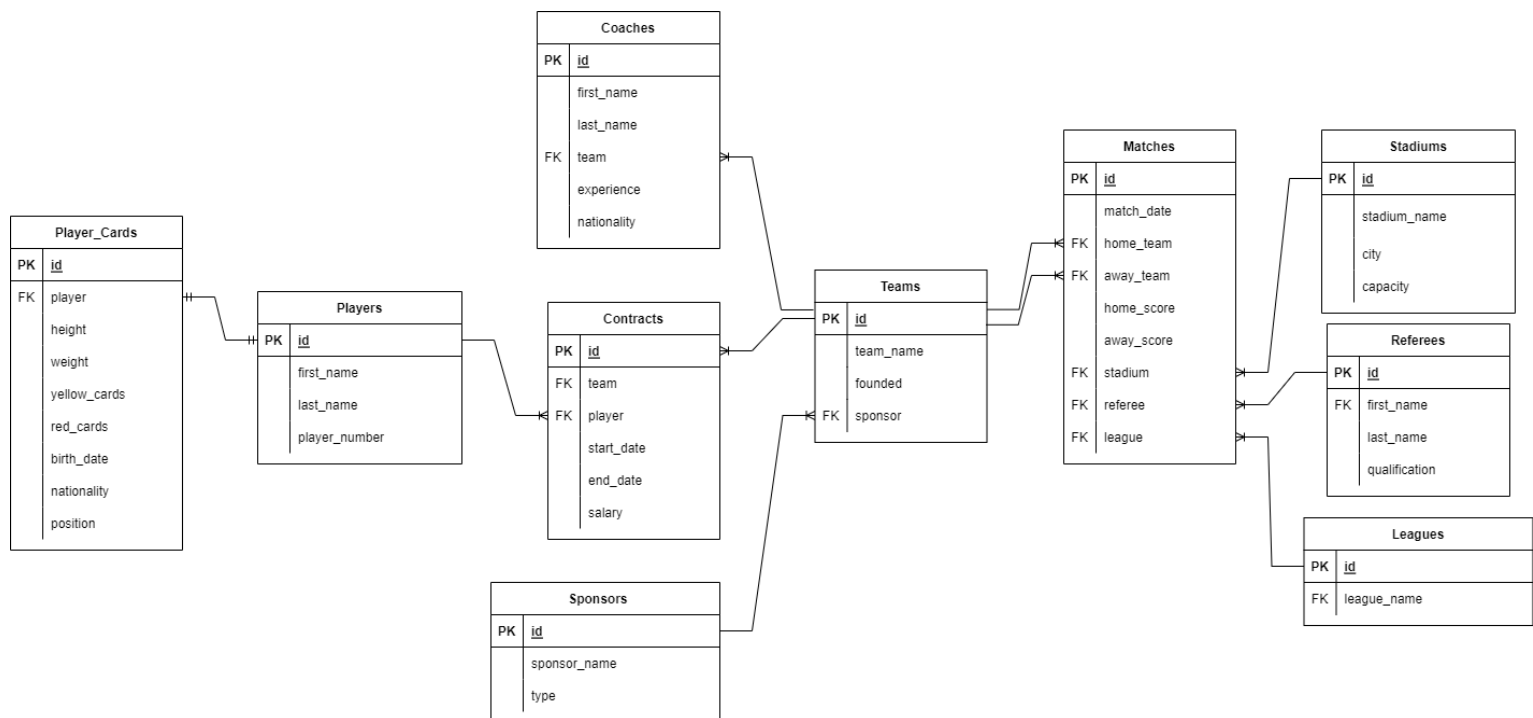


Рисунок 3.1 – ER-діаграма

4 РЕЛЯЦІЙНА МОДЕЛЬ БАЗИ ДАНИХ

4.1 Побудова необхідних відношень, визначення первинних та зовнішніх ключів

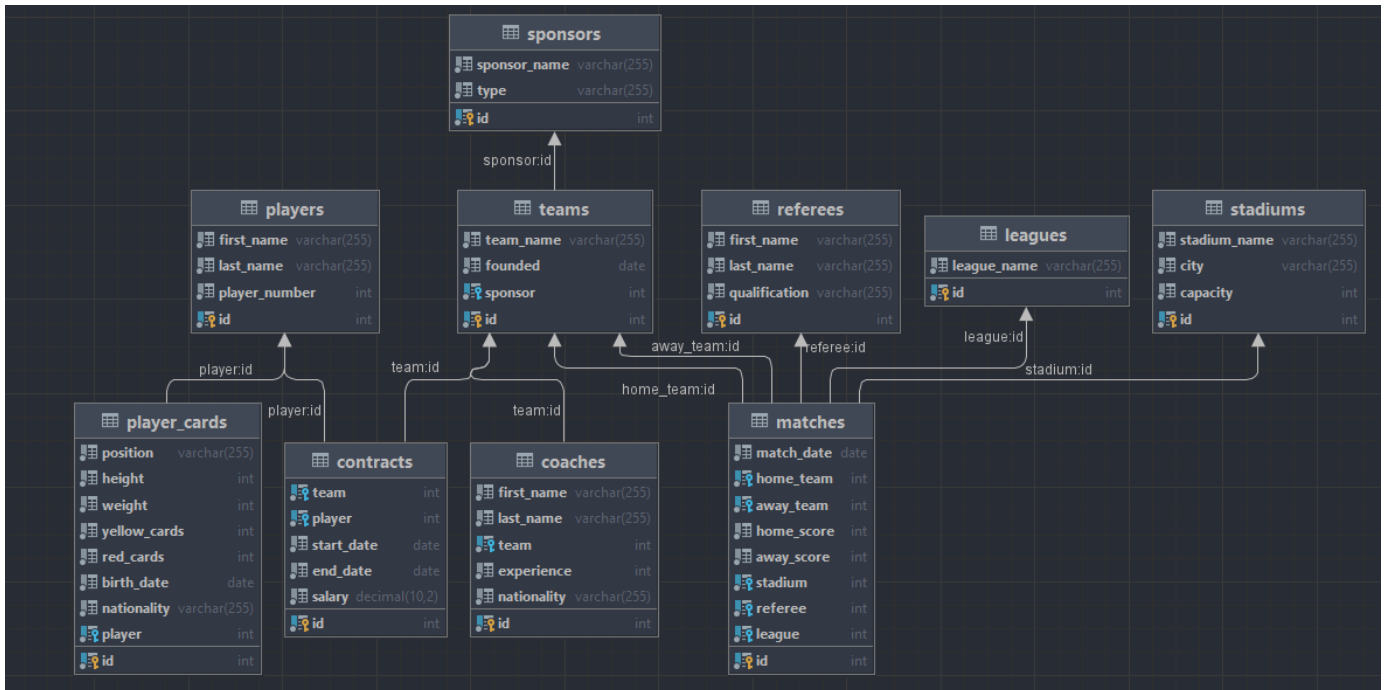


Рисунок 4.1– Реляційна схема бази даних

На даній схемі видно, що база даних знаходиться у 3 нормальній формі, адже всі поля таблиць декомпозовані, також всі атрибути таблиць функціонально повно залежать від первинного ключа, кожен неключовий атрибут не є транзитивно залежним від первинного ключа.

1. Обов'язкові атрибути таблиць мають обмеження NOT NULL, для запобігання помилок при роботі з даними.
2. Забезпечуються каскадні дії при видаленні зовнішніх ключів однієї з таблиць (ON DELETE CASCADE).

5 РЕАЛІЗАЦІЯ БАЗИ ДАНИХ

5.1 Створення бази даних

```
DROP DATABASE IF EXISTS football_association;
```

```
CREATE DATABASE football_association;
```

```
USE football_association;
```

```
CREATE TABLE Referees
```

```
(  
    id          INTEGER    NOT NULL AUTO_INCREMENT PRIMARY KEY,  
    first_name   VARCHAR(255) NOT NULL,  
    last_name    VARCHAR(255) NOT NULL,  
    qualification VARCHAR(255) NOT NULL  
);
```

```
CREATE TABLE Stadiums
```

```
(  
    id          INTEGER    NOT NULL AUTO_INCREMENT PRIMARY KEY,  
    stadium_name VARCHAR(255) NOT NULL,  
    city         VARCHAR(255) NOT NULL,  
    capacity     INTEGER    NOT NULL  
);
```

```
CREATE TABLE Sponsors
```

```
(  
    id          INTEGER    NOT NULL AUTO_INCREMENT PRIMARY KEY,  
    sponsor_name VARCHAR(255) NOT NULL,  
    type         VARCHAR(255) NOT NULL  
);
```

```
CREATE TABLE Teams
(
    id      INTEGER    NOT NULL AUTO_INCREMENT PRIMARY KEY,
    team_name VARCHAR(255) NOT NULL,
    founded DATE       NOT NULL,
    sponsor INTEGER    NOT NULL,
    FOREIGN KEY (sponsor) REFERENCES Sponsors (id) ON DELETE
CASCADE
);
```

```
CREATE TABLE Players
(
    id          INTEGER    NOT NULL AUTO_INCREMENT PRIMARY KEY,
    first_name  VARCHAR(255) NOT NULL,
    last_name   VARCHAR(255) NOT NULL,
    player_number INTEGER    NOT NULL,
    CHECK (player_number > 0 AND player_number < 100)
);
```

```
CREATE TABLE Player_Cards
(
    id          INTEGER    NOT NULL PRIMARY KEY,
    position    VARCHAR(255) NOT NULL,
    height      INTEGER    NOT NULL,
    weight      INTEGER    NOT NULL,
    yellow_cards INTEGER    NOT NULL,
    red_cards   INTEGER    NOT NULL,
    birth_date  DATE       NOT NULL,
    nationality VARCHAR(255) NOT NULL,
    player      INTEGER    NOT NULL,
```

```
FOREIGN KEY (player) REFERENCES Players (id) ON DELETE  
CASCADE,
```

```
    CHECK (position IN ('Forward', 'Defender', 'Midfielder', 'Goalkeeper'))  
);
```

```
CREATE TABLE Leagues
```

```
(  
    id      INTEGER NOT NULL AUTO_INCREMENT PRIMARY KEY,  
    league_name VARCHAR(255) NOT NULL  
);
```

```
CREATE TABLE Matches
```

```
(  
    id      INTEGER NOT NULL AUTO_INCREMENT PRIMARY KEY,  
    match_date DATE NOT NULL,  
    home_team INTEGER NOT NULL,  
    away_team INTEGER NOT NULL,  
    home_score INTEGER NOT NULL,  
    away_score INTEGER NOT NULL,  
    stadium  INTEGER NOT NULL,  
    referee  INTEGER NOT NULL,  
    league   INTEGER NOT NULL,  
    FOREIGN KEY (league) REFERENCES Leagues (id) ON DELETE  
CASCADE,  
    FOREIGN KEY (stadium) REFERENCES Stadiums (id) ON DELETE  
CASCADE,  
    FOREIGN KEY (home_team) REFERENCES Teams (id) ON DELETE  
CASCADE,  
    FOREIGN KEY (away_team) REFERENCES Teams (id) ON DELETE  
CASCADE,
```



```

FOREIGN KEY (referee) REFERENCES Referees (id) ON DELETE
CASCADE
);

```

```

CREATE TABLE Contracts
(
    id      INTEGER      NOT NULL AUTO_INCREMENT PRIMARY KEY,
    team    INTEGER      NOT NULL,
    player  INTEGER      NOT NULL,
    start_date DATE      NOT NULL,
    end_date DATE        NOT NULL,
    salary  DECIMAL(10, 2) NOT NULL,
    FOREIGN KEY (team) REFERENCES Teams (id) ON DELETE
CASCADE,
    FOREIGN KEY (player) REFERENCES Players (id) ON DELETE
CASCADE
);

```

```

CREATE TABLE Coaches
(
    id      INTEGER      NOT NULL AUTO_INCREMENT PRIMARY KEY,
    first_name VARCHAR(255) NOT NULL,
    last_name VARCHAR(255) NOT NULL,
    team    INTEGER      NOT NULL,
    experience INTEGER     NOT NULL,
    nationality VARCHAR(255) NOT NULL,
    FOREIGN KEY (team) REFERENCES Teams (id) ON DELETE CASCADE
);

```

5.2 Імпортування даних

Для імпортування даних у була використана можливість завантаження даних за .csv файлу. Було створено .csv файли для кожної з таблиць.

Перелік створених файлів зображено на рисунку:

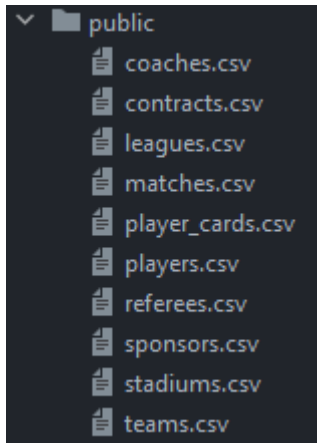


Рисунок 5.1 – Створені файли для заповнення

Також для меншої кількості даних було використано завантаження даних за допомогою запитів виду:

```
INSERT INTO players (first_name, last_name, player_number)  
VALUES ('TEST', 'TEST', 88);
```

6 СТВОРЕННЯ КОРИСТУВАЧІВ БАЗИ ДАНИХ

6.1 Рефери

```
DROP ROLE IF EXISTS referee;  
CREATE ROLE referee;  
GRANT SELECT, UPDATE (yellow_cards, red_cards) ON TABLE  
football_association.player_cards TO referee;  
GRANT SELECT, UPDATE, DELETE ON TABLE football_association.matches  
TO referee;
```

```
DROP USER IF EXISTS test_referee@localhost;  
CREATE USER test_referee@localhost IDENTIFIED BY 'referee';  
GRANT referee TO test_referee@localhost;
```

6.2 Вболівальник

```
DROP ROLE IF EXISTS fan;  
CREATE ROLE fan;  
GRANT SELECT ON football_association.match_with_teams TO fan;  
GRANT SELECT ON football_association.player_coach_contracts TO fan;
```

```
DROP USER IF EXISTS test_fan@localhost;  
CREATE USER test_fan@localhost IDENTIFIED BY 'fan';  
GRANT fan TO test_fan@localhost;
```

6.3 Менеджер команди

```
DROP ROLE IF EXISTS team_manager;  
CREATE ROLE team_manager;  
GRANT ALL ON football_association.contracts TO team_manager;  
GRANT ALL ON football_association.players TO team_manager;  
GRANT ALL ON football_association.coaches TO team_manager;  
GRANT ALL ON football_association.player_cards TO team_manager;
```

```
DROP USER IF EXISTS test_manager@localhost;
```

```
CREATE USER test_manager@localhost IDENTIFIED BY 'manager';  
GRANT team_manager TO test_manager@localhost;
```

7 SQL ЗАПИТИ

7.1 Створення тригерів на таблиці

Тригер, який не дозволяє змінити номер гравця на новий обраний номер гравцем, якщо цей номер зайнятий іншим гравцем з дійсним контрактом в межах цієї команди.

```
DROP TRIGGER IF EXISTS before_update_player_number;
```

```
DELIMITER $$
```

```
CREATE TRIGGER before_update_player_number
```

```
    BEFORE UPDATE
```

```
    ON players
```

```
    FOR EACH ROW
```

```
BEGIN
```

```
    DECLARE current_team, num_exists INT;
```

```
    SELECT team
```

```
    INTO current_team
```

```
    FROM contracts
```

```
    WHERE contracts.player = OLD.id
```

```
        AND end_date > CURDATE();
```

```
    SELECT COUNT(*)
```

```
    INTO num_exists
```

```
    FROM players
```

```
    WHERE players.player_number = NEW.player_number
```

```
        AND id IN (SELECT id
```

```
            FROM contracts
```

```
            WHERE contracts.team = current_team);
```

```
    IF num_exists > 0 AND OLD.player_number != NEW.player_number THEN
```

```
SIGNAL SQLSTATE '45000'
```

```
SET MESSAGE_TEXT = 'Player number already exists in team.';
```

```
END IF;
```

```
END $$
```

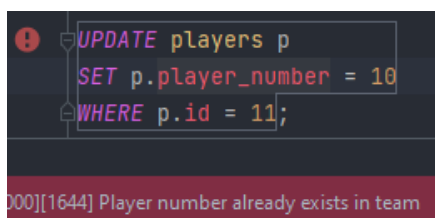
```
DELIMITER ;
```

Отримаємо список команди, де грає гравець з id = 11.



	id	first_name	last_name	player_number	team_name
1	1	Lionel	Messi	10	FC Dynamo
2	11	Luis	Suarez	9	FC Dynamo
3	21	Petr	Cech	77	FC Dynamo
4	31	Thierry	Henry	14	FC Dynamo
5	41	Dixie	Dean	17	FC Dynamo
6	51	Keylor	Navas	1	FC Dynamo
7	61	Zlatan	Kovac	65	FC Dynamo
8	71	Nemanja	Vidic	15	FC Dynamo
9	81	Santi	Cazorla	19	FC Dynamo
10	91	Jaap	Stam	4	FC Dynamo

При спробі зміни номера гравця за id = 11 на зайнятий номер отримаємо помилку.



```
UPDATE players p
SET p.player_number = 10
WHERE p.id = 11;
```

000][1644] Player number already exists in team

Тригер, який унеможливить додавання команди для певного спонсору, якщо цей спонсор співпрацює з двома командами.

```
DROP TRIGGER IF EXISTS before_insert_sponsor_team;
```

```
DELIMITER $$
```

```
CREATE TRIGGER before_insert_sponsor_team
```

```
BEFORE INSERT
```

```
ON teams
```

```
FOR EACH ROW
```

```
BEGIN
```

```

DECLARE sponsored_teams INT;

SELECT COUNT(*)

INTO sponsored_teams

FROM sponsors

      JOIN teams t on sponsors.id = t.sponsor

WHERE t.sponsor = NEW.sponsor;

IF sponsored_teams = 2 THEN

    SIGNAL SQLSTATE '45000'

        SET MESSAGE_TEXT = 'One sponsor can only work with a maximum of
two teams.';

END IF;

END $$

DELIMITER ;

```

Виконаємо запит завдяки якому побачимо з якою кількістю команд співпрацює кожен зі спонсорів.

	sponsor_id	sponsored_teams
1	1	2
2	2	1
3	3	1
4	4	2
5	5	1
6	6	1
7	7	2

При додаванні нової команди для співпраці зі спонсором id = 1 маємо помилку.

```

75
76 ! INSERT INTO teams (team_name, founded, sponsor)
77   VALUES ('TEST', '2000-01-01', 1);

```

[45000][1644] One sponsor can only work with a maximum of two teams.

Тригер, який не дозволяє видалити за бази федерації гравців, які мають дійсний контракт з будь-якою із команд.

```
DROP TRIGGER IF EXISTS before_delete_player;
```

```
DELIMITER $$
```

```
CREATE TRIGGER before_delete_player
```

```
    BEFORE DELETE
```

```
    ON Players
```

```
    FOR EACH ROW
```

```
BEGIN
```

```
    DECLARE contracts_exist INT;
```

```
    SELECT COUNT(*)
```

```
    INTO contracts_exist
```

```
    FROM contracts
```

```
    WHERE player = OLD.id
```

```
    AND end_date > CURDATE();
```

```
    IF contracts_exist > 0 THEN
```

```
        SIGNAL SQLSTATE '45000'
```

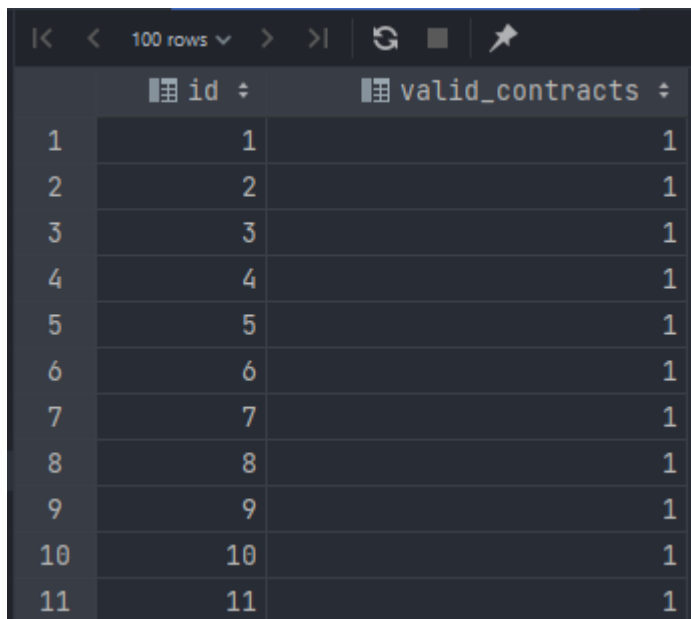
```
        SET MESSAGE_TEXT = 'Cannot delete player with valid contracts';
```

```
    END IF;
```

```
END $$
```

```
DELIMITER ;
```

Отримаємо дані про кожного гравця та наявність у нього дійсного контракту.



	id	valid_contracts
1	1	1
2	2	1
3	3	1
4	4	1
5	5	1
6	6	1
7	7	1
8	8	1
9	9	1
10	10	1
11	11	1

Отже, при спробі видалення гравця з $id = 1$ будемо мати помилку в результаті виконання триггеру.



```
124 DELETE
125 FROM players
126 WHERE id = 1;
```

[45000][1644] Cannot delete player with valid contracts

7.2 Створення процедур

Процедура, яка демонструє загальну кількість забитих голів певної команди в рамках певної ліги.

```
DROP PROCEDURE IF EXISTS get_team_goals_in_league;

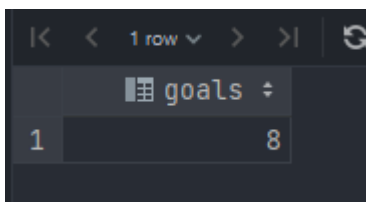
DELIMITER $$

CREATE PROCEDURE get_team_goals_in_league(IN team_id INT, IN league_id
INT)
BEGIN
    SELECT IF(SUM(m.home_score + m.away_score) IS NOT NULL,
        SUM(m.home_score + m.away_score),
        0) AS goals
    FROM matches m
        JOIN teams t ON (t.id = m.home_team OR t.id = m.away_team)
        JOIN leagues l ON l.id = m.league
    WHERE (t.id = team_id AND l.id = league_id);
END $$

DELIMITER ;

CALL get_team_goals_in_league(2, 2);
```

Загальна кількість забитих голів командою з id = 2 в рамках ліги з id = 2.



goals	
1	8

Процедура, яка демонструє гравців з дійсним контрактом в певній команді.

```
DROP PROCEDURE IF EXISTS get_players_with_valid_contract;

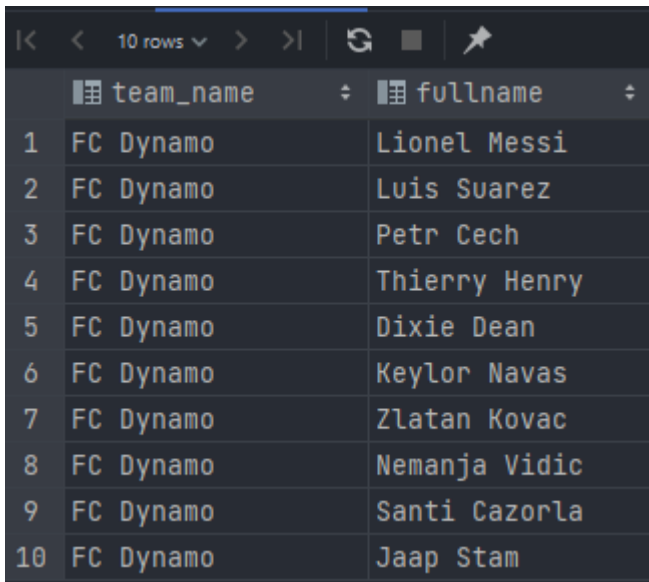
DELIMITER $$
```

```

CREATE PROCEDURE get_players_with_valid_contract(IN team_id INT)
BEGIN
    SELECT t.team_name,
           CONCAT(p.first_name, ' ', p.last_name) AS fullname
    FROM teams t
           JOIN contracts c on t.id = c.team
           JOIN players p on p.id = c.player
    WHERE t.id = team_id
           AND end_date > CURDATE();
END $$
DELIMITER ;
CALL get_players_with_valid_contract(1);

```

Гравці з команди id = 1 з дійсним контрактом.



	team_name	fullname
1	FC Dynamo	Lionel Messi
2	FC Dynamo	Luis Suarez
3	FC Dynamo	Petr Cech
4	FC Dynamo	Thierry Henry
5	FC Dynamo	Dixie Dean
6	FC Dynamo	Keylor Navas
7	FC Dynamo	Zlatan Kovac
8	FC Dynamo	Nemanja Vidic
9	FC Dynamo	Santi Cazorla
10	FC Dynamo	Jaap Stam

Процедура, яка змінює заробітню плату гравців певної команди на певний відсоток.

```

DROP PROCEDURE IF EXISTS update_team_salaries;
DELIMITER $$

```

```
CREATE PROCEDURE update_team_salaries(IN team_id INT, IN
percentage_to_change DECIMAL(5, 2), IN is_increase BOOLEAN)
```

```
BEGIN
```

```
    UPDATE contracts c
```

```
    SET c.salary = (1 + (2 * is_increase - 1) * (percentage_to_change / 100)) * c.salary
```

```
    WHERE c.team = team_id;
```

```
END $$
```

```
DELIMITER ;
```

Заробітня плата гравців команди з id = 5 до змін.

	id	team	player	start_date	end_date	salary
1	5	5	5	2018-01-01	2023-12-31	527216.26
2	15	5	15	2018-01-01	2023-12-31	636730.13
3	25	5	25	2018-01-01	2023-12-31	442881.03
4	35	5	35	2018-01-01	2023-12-31	331139.48
5	45	5	45	2018-01-01	2023-12-31	756483.49
6	55	5	55	2018-01-01	2023-12-31	295581.67
7	65	5	65	2018-01-01	2023-12-31	409759.21
8	75	5	75	2018-01-01	2023-12-31	706585.20
9	85	5	85	2018-01-01	2023-12-31	412686.70
10	95	5	95	2018-01-01	2023-12-31	471112.02

Викликаємо процедуру на збільшення заробітної плати на 7 відсотків.

```
CALL update_team_salaries(5, 7, TRUE);
```

Оновлені дані контрактів гравців даної команди.

	id	team	player	start_date	end_date	salary
1	5	5	5	2018-01-01	2023-12-31	564121.40
2	15	5	15	2018-01-01	2023-12-31	681301.24
3	25	5	25	2018-01-01	2023-12-31	473882.70
4	35	5	35	2018-01-01	2023-12-31	354319.24
5	45	5	45	2018-01-01	2023-12-31	809437.33
6	55	5	55	2018-01-01	2023-12-31	316272.39
7	65	5	65	2018-01-01	2023-12-31	438442.35
8	75	5	75	2018-01-01	2023-12-31	756046.16
9	85	5	85	2018-01-01	2023-12-31	441574.77
10	95	5	95	2018-01-01	2023-12-31	504089.86

Процедура, яка демонструє партнерів певного гравця, у яких дійсний контракт.

```
DROP PROCEDURE IF EXISTS get_team_by_player;

DELIMITER $$

CREATE PROCEDURE get_team_by_player(IN player_id INT)
BEGIN
    DECLARE team_id INT;

    SELECT DISTINCT team
    INTO team_id
    FROM players
        JOIN contracts c1 ON players.id = c1.player
    WHERE player = player_id
        AND c1.end_date > CURDATE();

    SELECT p.id, p.first_name, p.last_name, p.player_number, t.team_name
    FROM players p
        JOIN contracts c ON p.id = c.player
        JOIN teams t ON t.id = c.team
    WHERE t.id = team_id
        AND end_date > CURDATE();
END $$

DELIMITER ;

CALL get_team_by_player(11);
```

Склад команди, де грає гравець з id = 11.



	id	first_name	last_name	player_number	team_name
1	1	Lionel	Messi	10	FC Dynamo
2	11	Luis	Suarez	9	FC Dynamo
3	21	Petr	Cech	77	FC Dynamo
4	31	Thierry	Henry	14	FC Dynamo
5	41	Dixie	Dean	17	FC Dynamo
6	51	Keylor	Navas	1	FC Dynamo
7	61	Zlatan	Kovac	65	FC Dynamo
8	71	Nemanja	Vidic	15	FC Dynamo
9	81	Santi	Cazorla	19	FC Dynamo
10	91	Jaap	Stam	4	FC Dynamo

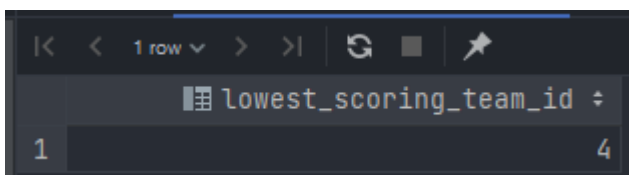
7.3 Створення функцій

Функція, яка повертає ідентифікатор команди з найменшою кількістю забитих голів.

```
DROP FUNCTION IF EXISTS get_lowest_scoring_team;
DELIMITER $$
CREATE FUNCTION get_lowest_scoring_team()
  RETURNS INT
  DETERMINISTIC
BEGIN
  DECLARE lowest_scoring_team INT;
  SELECT t.id
  INTO lowest_scoring_team
  FROM teams t
        JOIN matches m ON
        (t.id = m.home_team OR t.id = m.away_team)
  GROUP BY t.id
  ORDER BY SUM(m.home_score + m.away_score)
  LIMIT 1;
  RETURN lowest_scoring_team;
END $$
DELIMITER ;
```

Отримаємо найменш результативну команду в рамках усіх ліг.

```
SELECT get_lowest_scoring_team() AS lowest_scoring_team_id;
```



	lowest_scoring_team_id
1	4

Функція, яка повертає ідентифікатор найбільш оплачуваної команди за певним спонсором, з ким воно співпрацює.

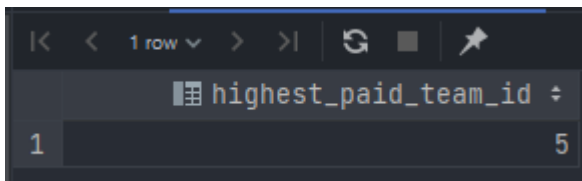
```
DROP FUNCTION IF EXISTS get_highest_paid_team_by_sponsor;
DELIMITER $$
```

```

CREATE FUNCTION get_highest_paid_team_by_sponsor(sponsor_id INT)
    RETURNS INT
    DETERMINISTIC
BEGIN
    DECLARE team_id INT;
    SELECT t.id
    INTO team_id
    FROM teams t
        JOIN contracts c ON t.id = c.team
    WHERE t.sponsor = sponsor_id
    GROUP BY t.id
    ORDER BY SUM(C.salary) DESC
    LIMIT 1;
    RETURN team_id;
END $$
DELIMITER ;
SELECT get_highest_paid_team_by_sponsor(5) AS highest_paid_team_id;

```

Найбільш оплачувана команда, яка співпрацює зі спонсором, ід якого 5.



	highest_paid_team_id
1	5

Функція, яка повертає гравця з найбільшою заробітною платою в рамках певної команди.

```

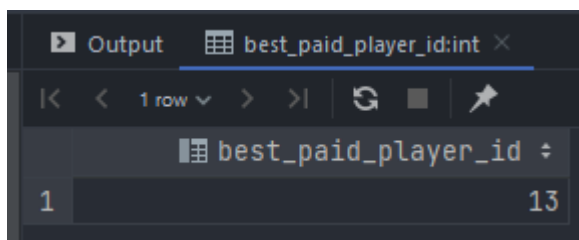
DROP FUNCTION IF EXISTS get_best_paid_player_for_team;
DELIMITER $$
CREATE FUNCTION get_best_paid_player_for_team(team_id INT)
    RETURNS INT
    DETERMINISTIC
BEGIN

```



```
DECLARE best_paid_player INT;  
SELECT p.id  
INTO best_paid_player  
FROM contracts c  
      JOIN players p ON c.player = p.id  
WHERE c.team = team_id  
ORDER BY c.salary DESC  
LIMIT 1;  
RETURN best_paid_player;  
END $$  
DELIMITER ;  
SELECT get_best_paid_player_for_team(8) AS best_paid_player_id;
```

Найбільша заробітня плата за контрактом у команді з id = 8 у наступного гравця:



The screenshot shows a database query output window with a dark theme. The title bar indicates the output is for the query 'best_paid_player_id:int'. The window shows a single row of results. The first column is labeled 'best_paid_player_id' and contains the value '1'. The second column is labeled '13'.

best_paid_player_id	13
1	13

7.4 Створення представлень

Представлення, яке демонструє біомедичні дані про гравців.

```
DROP VIEW IF EXISTS player_medcards;
```

```
CREATE OR REPLACE VIEW player_medcards AS
```

```
SELECT CONCAT(p.first_name, ' ', p.last_name)           AS
player_full_name,
```

```
    pc.height,
```

```
    pc.weight,
```

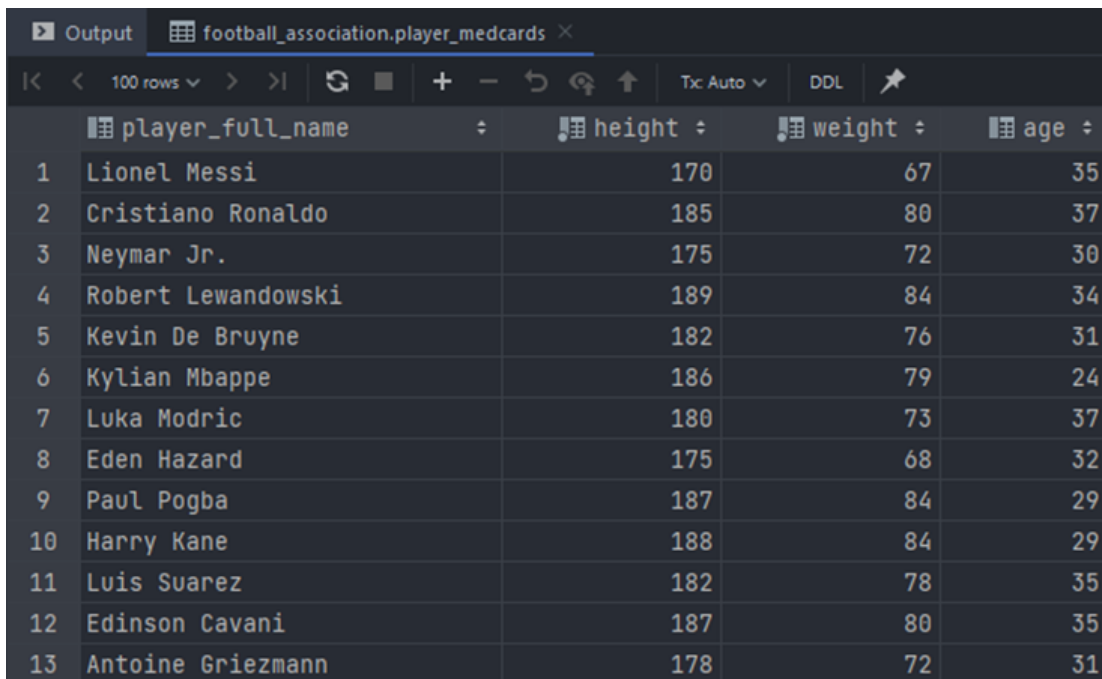
```
    DATE_FORMAT(FROM_DAYS(DATEDIFF(NOW(), pc.birth_date)), '%Y') +
0 AS age
```

```
FROM players p
```

```
    JOIN player_cards pc on p.id = pc.player;
```

```
SELECT *
```

```
FROM player_medcards;
```



The screenshot shows a database query result in a dark-themed interface. The window title is 'Output' and the tab is 'football_association.player_medcards'. The query result is displayed as a table with 4 columns: 'player_full_name', 'height', 'weight', and 'age'. There are 13 rows of data, numbered 1 to 13. The players listed are Lionel Messi, Cristiano Ronaldo, Neymar Jr., Robert Lewandowski, Kevin De Bruyne, Kylian Mbappe, Luka Modric, Eden Hazard, Paul Pogba, Harry Kane, Luis Suarez, Edinson Cavani, and Antoine Griezmann. The table has a dark background with light text for the column headers and row numbers, and alternating light and dark rows for the data.

	player_full_name	height	weight	age
1	Lionel Messi	170	67	35
2	Cristiano Ronaldo	185	80	37
3	Neymar Jr.	175	72	30
4	Robert Lewandowski	189	84	34
5	Kevin De Bruyne	182	76	31
6	Kylian Mbappe	186	79	24
7	Luka Modric	180	73	37
8	Eden Hazard	175	68	32
9	Paul Pogba	187	84	29
10	Harry Kane	188	84	29
11	Luis Suarez	182	78	35
12	Edinson Cavani	187	80	35
13	Antoine Griezmann	178	72	31

Представлення, яке демонструє дані контракту кожного з гравців разом з їх тренерами у більш змістованому вигляді.

```

DROP VIEW IF EXISTS player_coach_contracts;

CREATE OR REPLACE VIEW player_coach_contracts AS

SELECT CONCAT(p.first_name, ' ', p.last_name) AS player_full_name,

       t.team_name,

       CONCAT(co.first_name, ' ', co.last_name) AS coach_full_name,

       c.start_date,

       c.end_date,

       c.salary

FROM players p

       JOIN player_cards pc on p.id = pc.player

       JOIN contracts c on p.id = c.player

       JOIN teams t on t.id = c.team

       JOIN coaches co on t.id = co.team

ORDER BY p.id;

SELECT *

FROM player_coach_contracts;

```

	player_full_name	team_name	coach_full_name	start_date	end_date	salary
1	Lionel Messi	FC Dynamo	Pep Guardiola	2018-01-01	2023-12-31	729631.00
2	Lionel Messi	SC Tavriya	Marcelo Bielsa	2015-01-01	2017-12-31	882752.00
3	Lionel Messi	FC Dynamo	Mikel Arteta	2018-01-01	2023-12-31	729631.00
4	Cristiano Ronaldo	FC Shakhtar	Jurgen Klopp	2018-01-01	2023-12-31	365103.00
5	Cristiano Ronaldo	SC Olimpik	Zinedine Zidane	2015-01-01	2017-12-31	118902.00
6	Cristiano Ronaldo	FC Shakhtar	Antonio Conte	2018-01-01	2023-12-31	365103.00
7	Neymar Jr.	FC Dnipro	Maurizio Sarri	2018-01-01	2023-12-31	270248.00
8	Neymar Jr.	FC Vorskla	Julen Lopetegui	2015-01-01	2017-12-31	175888.00
9	Neymar Jr.	FC Dnipro	Unai Emery	2018-01-01	2023-12-31	270248.00
10	Neymar Jr.	FC Vorskla	Ralph Hasenhuttl	2015-01-01	2017-12-31	175888.00
11	Robert Lewandowski	FC Zorya	Eddie Howe	2018-01-01	2023-12-31	878473.76
12	Robert Lewandowski	SC Chornomorets	Nuno Espirito Santo	2015-01-01	2017-12-31	230655.00
13	Robert Lewandowski	FC Zorya	Ole Gunnar	2018-01-01	2023-12-31	878473.76

Представлення, яке демонструє результати зіграних матчів у більш явному вигляді.

```
DROP VIEW IF EXISTS match_with_teams;
```

```
CREATE OR REPLACE VIEW match_with_teams AS
```

```
SELECT DISTINCT t_home.team_name AS home_team,
```

```
CONCAT(m.home_score, ' : ', m.away_score) AS score,
```

```
t_away.team_name AS away_team
```

```
FROM matches m
```

```
JOIN teams t_home ON m.home_team = t_home.id
```

```
JOIN teams t_away ON m.away_team = t_away.id;
```

```
SELECT *
```

```
FROM match_with_teams;
```

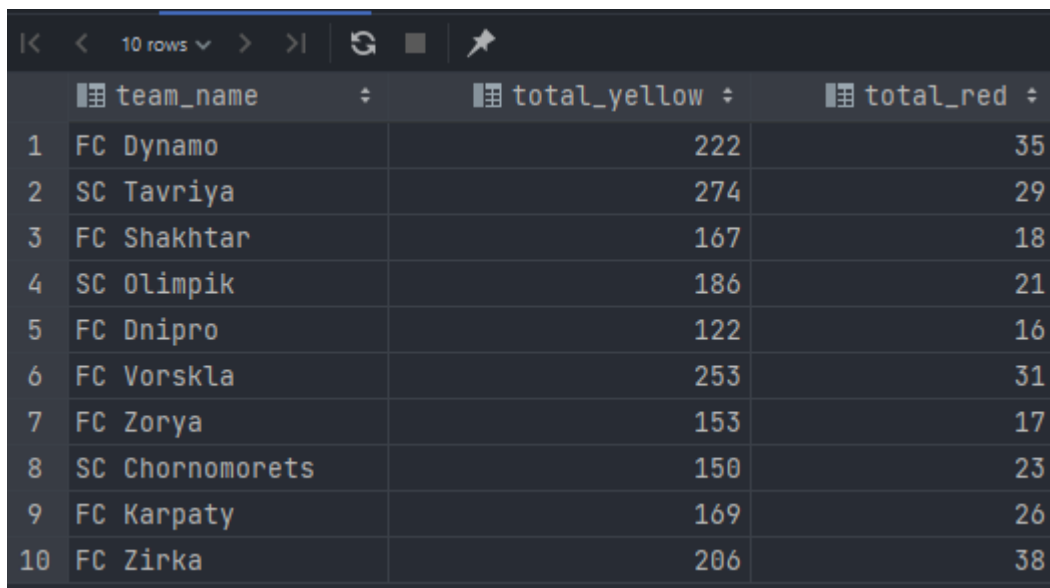


	home_team	score	away_team
1	FC Dynamo	4 : 1	FC Shakhtar
2	FC Dynamo	2 : 2	FC Dnipro
3	FC Dynamo	1 : 0	FC Zorya
4	FC Dynamo	0 : 1	FC Karpaty
5	FC Dynamo	2 : 2	SC Chornomorets
6	FC Dynamo	1 : 1	FC Vorskla
7	FC Dynamo	2 : 1	SC Olimpik
8	FC Dynamo	2 : 0	SC Tavriya
9	FC Dynamo	1 : 1	FC Zirka
10	FC Shakhtar	0 : 3	FC Dnipro
11	FC Shakhtar	1 : 1	FC Zorya
12	FC Shakhtar	1 : 1	FC Karpaty
13	FC Shakhtar	1 : 0	SC Chornomorets

7.5 Створення різних запитів

Цей запит демонструє назву кожної команди, загальну кількість жовтих і червоних карток, отриманих гравцями команди.

```
SELECT t.team_name,
       SUM(pc.yellow_cards) AS total_yellow,
       SUM(pc.red_cards)   AS total_red
FROM teams t
      JOIN contracts c on t.id = c.team
      JOIN players p on p.id = c.player
      JOIN player_cards pc on p.id = pc.player
GROUP BY team;
```



	team_name	total_yellow	total_red
1	FC Dynamo	222	35
2	SC Tavriya	274	29
3	FC Shakhtar	167	18
4	SC Olimpik	186	21
5	FC Dnipro	122	16
6	FC Vorskla	253	31
7	FC Zorya	153	17
8	SC Chornomorets	150	23
9	FC Karpaty	169	26
10	FC Zirka	206	38

Цей запит демонструє середній зріст гравців для кожної позиції.

```
SELECT pc.position,
       AVG(pc.height) AS average_height,
       AVG(pc.weight) AS average_weight
FROM players p
      JOIN player_cards pc on p.id = pc.player
GROUP BY pc.position;
```

	position	average_height	average_weight
1	Forward	181.6857	76.4000
2	Defender	187.8750	82.7813
3	Midfielder	179.2692	73.0385
4	Goalkeeper	190.0000	82.5714

Цей запит демонструє назву кожної команди та максимальну зарплату гравця в цій команді.

```
SELECT t.team_name, MAX(c.salary) AS max_salary
```

```
FROM players p
```

```
JOIN contracts c on p.id = c.player
```

```
JOIN teams t on t.id = c.team
```

```
GROUP BY t.team_name;
```

	team_name	max_salary
1	FC Dynamo	935847.00
2	FC Shakhtar	993843.00
3	FC Dnipro	974659.00
4	FC Zorya	917818.31
5	FC Karpaty	1010994.63
6	SC Chornomorets	999972.00
7	FC Vorskla	916310.00
8	SC Olimpik	971723.00
9	SC Tavriya	882752.00
10	FC Zirka	925736.00

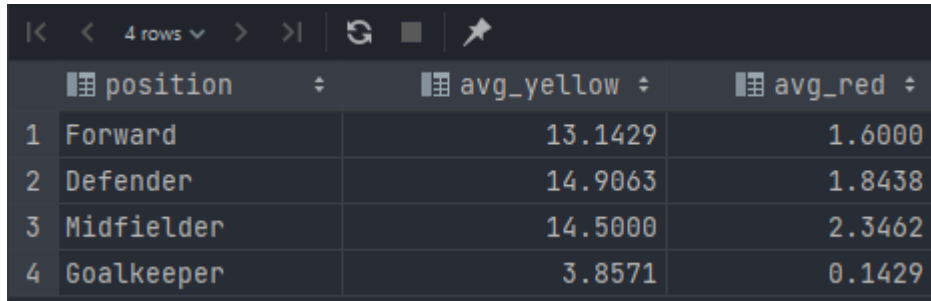
Цей запит демонструє середню кількість жовтих і червоних карток для кожної позиції.

```
SELECT pc.position,
```

```
AVG(pc.yellow_cards) AS avg_yellow,
```

```
AVG(pc.red_cards) AS avg_red
```

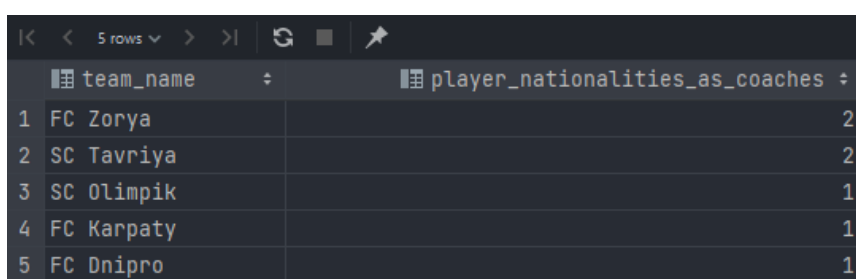
```
FROM players p
      JOIN player_cards pc on p.id = pc.player
GROUP BY position;
```



	position	avg_yellow	avg_red
1	Forward	13.1429	1.6000
2	Defender	14.9063	1.8438
3	Midfielder	14.5000	2.3462
4	Goalkeeper	3.8571	0.1429

Цей запит демонструє назву команди та кількість гравців у кожній команді, які мають ту саму національність, що й тренер команди, у якій мають дійний контракт.

```
SELECT t.team_name,
      COUNT(pc.nationality) AS player_nationalities_as_coaches
FROM players p
      JOIN player_cards pc on p.id = pc.player
      JOIN contracts c ON p.id = c.player
      JOIN teams t ON t.id = c.team
      JOIN coaches co ON co.team = t.id
WHERE c.end_date > CURDATE()
      AND pc.nationality = co.nationality
GROUP BY t.team_name
ORDER BY player_nationalities_as_coaches DESC;
```



	team_name	player_nationalities_as_coaches
1	FC Zorya	2
2	SC Tavriya	2
3	SC Olimpik	1
4	FC Karpaty	1
5	FC Dnipro	1

Цей запит демонструє назву команди та гравців у кожній команді, які мають ту саму національність, що й тренер команди, і наразі мають контракт.

```
SELECT CONCAT(p.first_name, ' ', p.last_name) AS player_fullname,
        CONCAT(co.first_name, ' ', co.last_name) AS coach_fullname,
        pc.nationality,
        co.nationality,
        t.team_name
FROM players p
        JOIN player_cards pc on p.id = pc.player
        JOIN contracts c ON p.id = c.player
        JOIN teams t ON t.id = c.team
        JOIN coaches co ON co.team = t.id
WHERE c.end_date > CURDATE()
AND pc.nationality = co.nationality;
```

	player_fullname	coach_fullname	pc.nationality	co.nationality	team_name
1	Steven Gerrard	Eddie Howe	England	England	FC Zorya
2	Barkley Ross	Marcelo Bielsa	Argentina	Argentina	SC Tavriya
3	Gonzalo Higuain	Marcelo Bielsa	Argentina	Argentina	SC Tavriya
4	Aymeric Laporte	Zinedine Zidane	France	France	SC Olimpik
5	Jerome Boateng	Thomas Tuchel	Germany	Germany	FC Karpaty
6	John Terry	Unai Emery	Spain	Spain	FC Dnipro
7	Sol Campbell	Eddie Howe	England	England	FC Zorya

Цей запит демонструє середню зарплату гравця та повне ім'я кожного судді, який обслуговував матч за участю команди, яка мала гравця за контрактом на час матчу.

```
SELECT ROUND(AVG(c.salary), 2) AS average_player_salary,
        CONCAT(r.first_name, ' ', r.last_name) AS referee_fullname
FROM players p
        JOIN player_cards pc on p.id = pc.player
```


JOIN contracts c on p.id = c.player

JOIN teams t on t.id = c.team

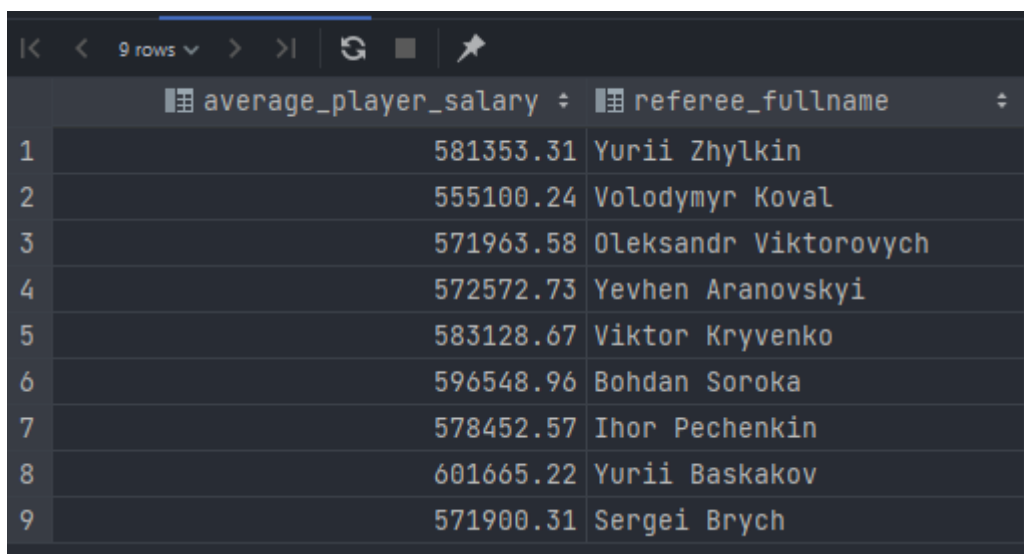
JOIN matches m on (t.id = m.away_team OR m.home_team = t.id)

JOIN referees r on m.referee = r.id

WHERE c.end_date > m.match_date

AND c.start_date < m.match_date

GROUP BY r.id;



	average_player_salary	referee_fullname
1	581353.31	Yurii Zhylkin
2	555100.24	Volodymyr Koval
3	571963.58	Oleksandr Viktorovych
4	572572.73	Yevhen Aranovskyi
5	583128.67	Viktor Kryvenko
6	596548.96	Bohdan Soroka
7	578452.57	Ihor Pechenkin
8	601665.22	Yurii Baskakov
9	571900.31	Sergei Brych

Цей запит демонструє назву стадіону, назву команди та кількість матчів, зіграних на кожному стадіоні кожною командою.

SELECT s.stadium_name,

t.team_name AS team_name,

COUNT(DISTINCT m.id) AS match_count

FROM stadiums s

JOIN matches m ON m.stadium = s.id

JOIN teams t ON (t.id = m.home_team OR t.id = m.away_team)

JOIN contracts c ON t.id = c.team

JOIN players p ON p.id = c.player

```

WHERE c.end_date > m.match_date

AND c.start_date < m.match_date

GROUP BY s.id, t.id

ORDER BY s.id, match_count DESC;

```

	stadium_name	team_name	match_count
1	NSC Olimpiyskiy	FC Shakhtar	2
2	NSC Olimpiyskiy	FC Dynamo	1
3	NSC Olimpiyskiy	FC Karpaty	1
4	NSC Olimpiyskiy	SC Chornomorets	1
5	NSC Olimpiyskiy	SC Tavriya	1
6	Olympic Stadium	FC Dynamo	1
7	Olympic Stadium	FC Shakhtar	1
8	Olympic Stadium	FC Dnipro	1
9	Olympic Stadium	FC Karpaty	1
10	Olympic Stadium	FC Vorskla	1
11	Olympic Stadium	FC Zirka	1
12	Arena Lviv	FC Zorya	2
13	Arena Lviv	FC Dynamo	1

Цей запит демонструє назву команди та загальну кількість голів, забитих кожною командою вдома чи на виїзді.

```

SELECT t.team_name,

SUM(IF(t.id = m.home_team, m.home_score, 0)) AS home_goals,

SUM(IF(t.id = m.away_team, m.away_score, 0)) AS away_goals

FROM teams t

JOIN matches m ON (t.id = m.home_team OR t.id = m.away_team)

GROUP BY t.id;

```

	team_name	home_goals	away_goals
1	FC Dynamo	15	0
2	FC Shakhtar	14	1
3	FC Dnipro	12	5
4	FC Zorya	10	2
5	FC Karpaty	11	5
6	SC Chornomorets	8	5
7	FC Vorskla	7	6
8	SC Olimpik	6	13
9	SC Tavriya	2	11
10	FC Zirka	0	11

Цей запит демонструє назву спонсора та назву ліги для кожного спонсора, пов'язаного з командою в певній лізі.

```
SELECT DISTINCT sp.sponsor_name, l.league_name
```

```
FROM sponsors sp
```

```
JOIN teams t ON sp.id = t.sponsor
```

```
JOIN matches m on (t.id = m.away_team OR m.home_team = t.id)
```

```
JOIN leagues l on l.id = m.league
```

```
ORDER BY sp.sponsor_name;
```

	sponsor_name	league_name
1	Adidas	Ukrainian Premier League
2	Adidas	Ukrainian First League
3	Adidas	Ukrainian Second League
4	Balenciaga	Ukrainian Premier League
5	Balenciaga	Ukrainian First League
6	Balenciaga	Ukrainian Second League
7	Coca-Cola	Ukrainian Premier League
8	Coca-Cola	Ukrainian First League
9	Coca-Cola	Ukrainian Second League
10	Heineken	Ukrainian Premier League
11	Heineken	Ukrainian First League
12	Heineken	Ukrainian Second League
13	Nike	Ukrainian Premier League

Цей запит демонструє тип спонсора, назву команди та середнього гравця вік для кожної команди, згрупований за назвою команди та типом спонсора.

```
SELECT s.type AS sponsor_type,
       t.team_name,
       ROUND(AVG(DATE_FORMAT(FROM_DAYS(DATEDIFF(NOW(),
pc.birth_date)), '%Y')), 2) AS average_age
FROM sponsors s
      JOIN teams t on s.id = t.sponsor
      JOIN contracts c on t.id = c.team
      JOIN players p on c.player = p.id
      JOIN player_cards pc on p.id = pc.player
GROUP BY t.team_name, sponsor_type
ORDER BY average_age;
```

	sponsor_type	team_name	average_age
1	Fashion	SC Chornomorets	35.92
2	Energy Drink	FC Karpaty	36.15
3	Beverage	SC Olimpik	37.38
4	Fashion	FC Zorya	40.86
5	Beverage	FC Dnipro	41
6	Beer	FC Shakhtar	42.46
7	Sportswear	FC Zirka	43.62
8	Sportswear	FC Vorskla	44
9	Beverage	SC Tavriya	44.29
10	Sportswear	FC Dynamo	45

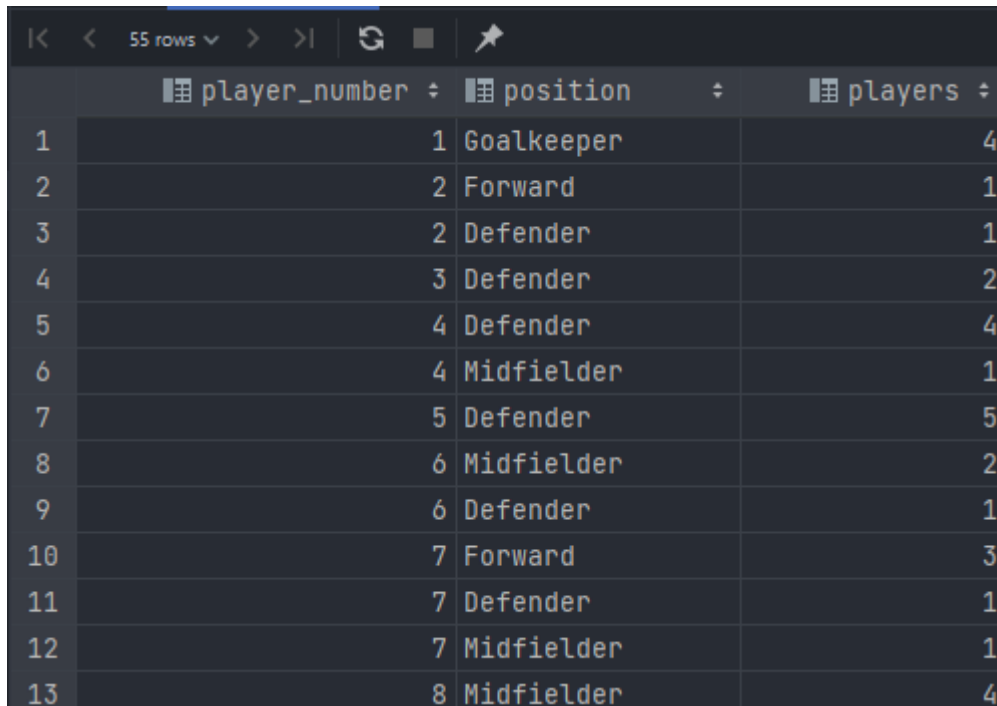
Цей запит демонструє номер гравця, позицію та кількість гравців із кожною комбінацією номера гравця та позиції.

```
SELECT player_number,
       pc.position,
```

```

COUNT(player_number) AS players
FROM players
    JOIN player_cards pc on players.id = pc.player
GROUP BY player_number, pc.position
ORDER BY player_number, players DESC;

```



	player_number	position	players
1	1	Goalkeeper	4
2	2	Forward	1
3	2	Defender	1
4	3	Defender	2
5	4	Defender	4
6	4	Midfielder	1
7	5	Defender	5
8	6	Midfielder	2
9	6	Defender	1
10	7	Forward	3
11	7	Defender	1
12	7	Midfielder	1
13	8	Midfielder	4

Цей запит демонструє позицію та середню зарплату гравця для кожної позиції, упорядковану за середньою зарплатою в порядку спадання.

```

SELECT pc.position,
    ROUND(AVG(c.salary), 2) AS avg_player_salary
FROM players p
    JOIN player_cards pc on p.id = pc.player
    JOIN contracts c on p.id = c.player
GROUP BY pc.position
ORDER BY avg_player_salary DESC;

```

	position	avg_player_salary
1	Midfielder	688382.89
2	Goalkeeper	587837.65
3	Forward	587637.40
4	Defender	491205.25

Цей запит демонструє назву команди, середню зарплату гравця та середній досвід тренера для кожної команди, впорядковані за середньою зарплатою гравця в порядку спадання.

```
SELECT team_name,
```

```
       ROUND(AVG(salary), 2) AS average_players_salary,
```

```
       ROUND(AVG(experience), 1) AS average_coaches_exp
```

```
FROM coaches
```

```
     JOIN teams t on coaches.team = t.id
```

```
     JOIN contracts c on t.id = c.team
```

```
GROUP BY team_name
```

```
ORDER BY average_players_salary DESC;
```

	team_name	average_players_salary	average_coaches_exp
1	SC Chornomorets	645752.54	4.0
2	SC Tavriya	630336.50	9.0
3	FC Zirka	628636.92	8.0
4	FC Zorya	624125.10	5.5
5	FC Dynamo	609948.15	9.0
6	FC Karpaty	578193.96	8.0
7	FC Dnipro	558781.38	5.5
8	FC Shakhtar	539823.00	10.0
9	SC Olimpik	517202.15	12.0
10	FC Vorskla	511798.00	5.5

Цей запит демонструє назву ліги та середню місткість стадіону для кожної ліги, упорядковану за середньою місткістю в порядку спадання.

```

SELECT l.league_name,
       CAST(AVG(s.capacity) AS FLOAT) AS avg_stadium_capacity
FROM matches m
       JOIN leagues l on m.league = l.id
       JOIN stadiums s on m.stadium = s.id
GROUP BY l.id
ORDER BY avg_stadium_capacity DESC;

```

	league_name	avg_stadium_capacity
1	Ukrainian First League	38000
2	Ukrainian Premier League	37800
3	Ukrainian Second League	33800

Цей запит демонструє дату матчу, назву домашньої команди, назву команди на виїзді та суму зарплат гравців домашньої та виїзної команд для кожного матчу, де команда господарів має вищу суму зарплати гравця.

```

SELECT m.match_date,
       ht.team_name      AS home_team_name,
       at.team_name      AS away_team_name,
       (SELECT SUM(c.salary)
        FROM contracts c
              JOIN players p ON c.player = p.id
              WHERE c.team = m.home_team) AS home_salary_sum,
       (SELECT SUM(c.salary)
        FROM contracts c
              JOIN players p ON c.player = p.id
              WHERE c.team = m.away_team) AS away_salary_sum

```

```

FROM matches m

    JOIN teams ht ON m.home_team = ht.id

    JOIN teams at ON m.away_team = at.id

HAVING home_salary_sum > away_salary_sum;

```

	match_date	home_team_name	away_team_name	home_salary_sum	away_salary_sum
1	2022-01-01	FC Dynamo	FC Shakhtar	7929326.00	7017699.00
2	2022-01-02	FC Dynamo	FC Dnipro	7929326.00	7264158.00
3	2022-01-04	FC Dynamo	FC Karpaty	7929326.00	7516521.43
4	2022-01-06	FC Dynamo	FC Vorskla	7929326.00	6653374.00
5	2022-01-07	FC Dynamo	SC Olimpik	7929326.00	6723628.00
6	2022-03-03	FC Shakhtar	FC Vorskla	7017699.00	6653374.00
7	2022-03-04	FC Shakhtar	SC Olimpik	7017699.00	6723628.00
8	2022-03-10	FC Dnipro	FC Vorskla	7264158.00	6653374.00
9	2022-03-11	FC Dnipro	SC Olimpik	7264158.00	6723628.00
10	2022-03-14	FC Zorya	FC Karpaty	8737751.39	7516521.43
11	2022-07-15	FC Zorya	SC Chornomorets	8737751.39	8394783.00
12	2022-07-16	FC Zorya	FC Vorskla	8737751.39	6653374.00

Цей запит демонструє назву команди, кількість гравців без червоних або жовтих карток і середню зарплату гравця для кожної команди, у всіх гравців якої немає червоних або жовтих карток.

```

SELECT t.team_name,

    COUNT(p.id)          AS total_players_without_card,

    ROUND(AVG(c.salary), 2) AS avg_player_salary

FROM teams t

    JOIN contracts c ON t.id = c.team

    JOIN players p ON c.player = p.id

    JOIN player_cards pc ON p.id = pc.player

WHERE pc.red_cards = 0

    AND pc.yellow_cards = 0

GROUP BY t.id

ORDER BY avg_player_salary DESC;

```

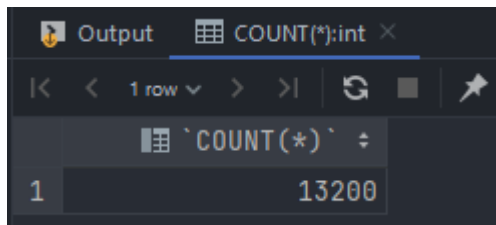

	team_name	total_players_without_card	avg_player_salary
1	FC Zirka	1	925736.00
2	FC Karpaty	1	756046.16
3	FC Shakhtar	1	665035.00
4	FC Zorya	2	579675.87
5	FC Dynamo	1	381401.00
6	SC Chornomorets	1	351063.00
7	FC Dnipro	1	264276.00
8	FC Vorskla	1	120342.00

7.6 Створення індексів

Створимо таблицю для кращої демонстрації роботи індексів. Таблиця буде декартовим добутком двох таблиць, завдяки чому буде мати в рази більше записів, ніж таблицях.

```
DROP TABLE IF EXISTS index_test;
CREATE TABLE index_test AS
SELECT p.id, p.first_name, p. last_name, p.player_number, c.team, c.start_date,
c.end_date, c.salary
FROM players p
      CROSS JOIN contracts c;
```

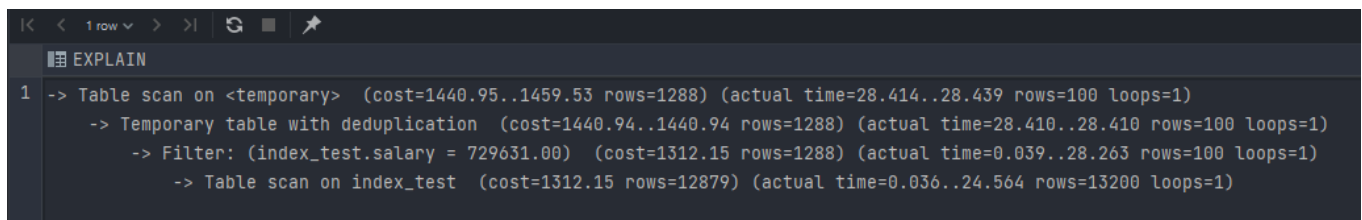
```
SELECT COUNT(*)
FROM index_test;
```



COUNT(*)	
1	13200

Статистичні дані виконання простого запиту без створеного індексу.

```
EXPLAIN ANALYZE
SELECT DISTINCT id, last_name, last_name, team
FROM index_test
WHERE salary = 729631.00;
```

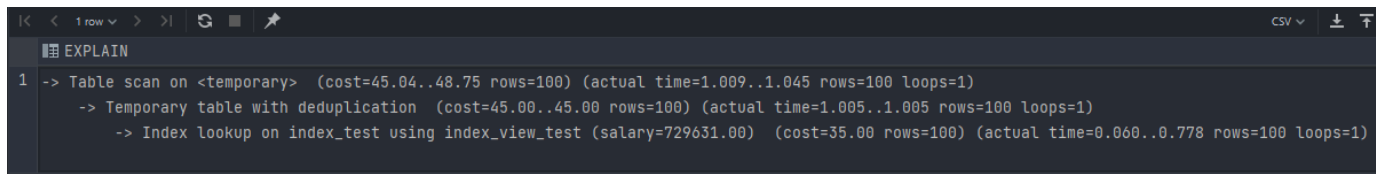


EXPLAIN	
1	-> Table scan on <temporary> (cost=1440.95..1459.53 rows=1288) (actual time=28.414..28.439 rows=100 loops=1) -> Temporary table with deduplication (cost=1440.94..1440.94 rows=1288) (actual time=28.410..28.410 rows=100 loops=1) -> Filter: (index_test.salary = 729631.00) (cost=1312.15 rows=1288) (actual time=0.039..28.263 rows=100 loops=1) -> Table scan on index_test (cost=1312.15 rows=12879) (actual time=0.036..24.564 rows=13200 loops=1)

Створимо індекс на поле за яким здійснюємо пошук створеної тестової таблиці.

```
CREATE INDEX index_view_test
ON index_test(salary);
```

Статистичні дані виконання такого ж запиту після створення індексу.



	EXPLAIN
1	-> Table scan on <temporary> (cost=45.04..48.75 rows=100) (actual time=1.009..1.045 rows=100 loops=1) -> Temporary table with deduplication (cost=45.00..45.00 rows=100) (actual time=1.005..1.005 rows=100 loops=1) -> Index lookup on index_test using index_view_test (salary=729631.00) (cost=35.00 rows=100) (actual time=0.060..0.778 rows=100 loops=1)

Можно побачити, що продуктивність виконання запиту збільшилася приблизно в 28 разів. З даного тестування можна зрозуміти переваги використання індексів в різних базах даних.

ВИСНОВОК

Перед виконанням завдання по створенню бази даних по підтримці діяльності федерації футболу країни було спроектовану майбутню базу даних, виокремленні основні сутності та зв'язки між ними. Були сформульовані бізнес-правила та вимоги, які потрібно було реалізувати у роботі.

Отже, у даній роботі було успішно розроблено та впроваджено базу даних для підтримки діяльності футбольної федерації країни. Було створено 10 різних таблиць та визначені зв'язки між ними за допомогою зовнішніх ключів і каскадних дій. Були закріплені навички створювання та використання різних об'єктів бази даних, такі як тригери, функції та процедури, створення складних запитів, яке несуть практичне значення для потенційних користувачів сховища даних.

ПЕРЕЛІК ПОСИЛАНЬ

1. <https://dev.mysql.com/doc/>
2. <https://www.jetbrains.com/help/datagrip/meet-the-product.html>