МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ ІМЕНІ ІГОРЯ СІКОРСЬКОГО»
Факультет інформатики та обчислювальної техніки
Кафедра інформатики та програмної інженерії

Практикум №1-2
з курсу «Аналіз даних в інформаційнних системах»
на тему: «Створення процедур завантаження даних»

Викладач:
Олійник Ю.О.

Виконала:
студентка 2 курсу
Групи ІП-21
Скрипець Ольга
ФІОТ

Київ-2024

Для виконання лабораторної роботи було обрано датасет на сайті https://www.kaggle.com/, що містить декілька таблиць, пов’язаних із футболом. У подальшій роботі було взято 4 таблиці: стадіон (stadium), матчі (matches), команди (teams), тренери (managers).

1. Таблиця "stadiums" містить дані про стадіони, такі як їх назва, розташування (місто та країна), а також місткість.
2. Таблиця "matches" містить дані про всі матчі протягом певного періоду, зазначаючи сезон, дату та час, назви домашньої та гостьової команд, стадіон, кількість голів, чи була серія пенальті та відвідування матчу глядачами.
3. Таблиця "teams" містить інформацію про всі команди, що брали участь у турнірі, зокрема назву, країну походження та домашній стадіон.
4. Таблиця "managers" містить інформацію про тренерів, включаючи їх національність, дату народження та команду, яку вони керують.

Посилання на сам датасет, де були взяті дані: https://www.kaggle.com/datasets/cbxkgl/uefa-champions-league-2016-2022-data

Опис джерел даних

Ось таблиця Stadiums.csv, яка містить назви 4 полів та їх призначення:

Назва поля	Призначення
name	назва стадіону
city	місто, в якому знаходиться стадіон
country	країна, в якій знаходиться стадіон
capacity	місткість стадіону

Ось таблиця Teams.csv, яка містить назви 3 полів та їх призначення:

Назва поля	Призначення
team_name	назва команди
country	країна походження команди
home_stadium	домашній стадіон команди

Ось таблиця Managers.csv, яка містить назви 5 полів та їх призначення:

Назва поля	Призначення
first_name	ім’я тренера
last_name	прізвище тренера
nationality	національність тренера
date_of_birth	дата народження тренера
team	команда, яку він керує

Ось таблиця Matches.csv, яка містить назви 10 полів та їх призначення:

Назва поля	**Призначення**
match_id	ідентифікатор матчу
season	сезон, в якому відбувся матч
date_time	дата та час проведення матчу
home_team	назва домашньої команди
away_team	назва гостьової команди
stadium	стадіон, на якому відбувся матч
home_team_score	кількість голів, забитих домашньою командою
away_team_score	кількість голів, забитих гостьовою командою
penalty_shoot_out	чи була серія пенальті після матчу
attendance	кількість глядачів, які відвідали матч

Редагування даних

Я відредагувала csv файли для зручної подальшої обробки та аналізу даних. Стандартний формат дати (YYYY-MM-DD) є більш зручним для обробки та аналізу, тому я змінила його на такий.

Також я оновила ідентифікатори матчів, видаляючи «mt» перші два символи, для зручнішого формату ідентифікаторів матчів.

В результаті цих змін отримала оновлені CSV-файли, які тепер можна легко аналізувати. Формат дати та ідентифікаторів матчів був уніфікований. Це полегшує розуміння та аналіз даних.

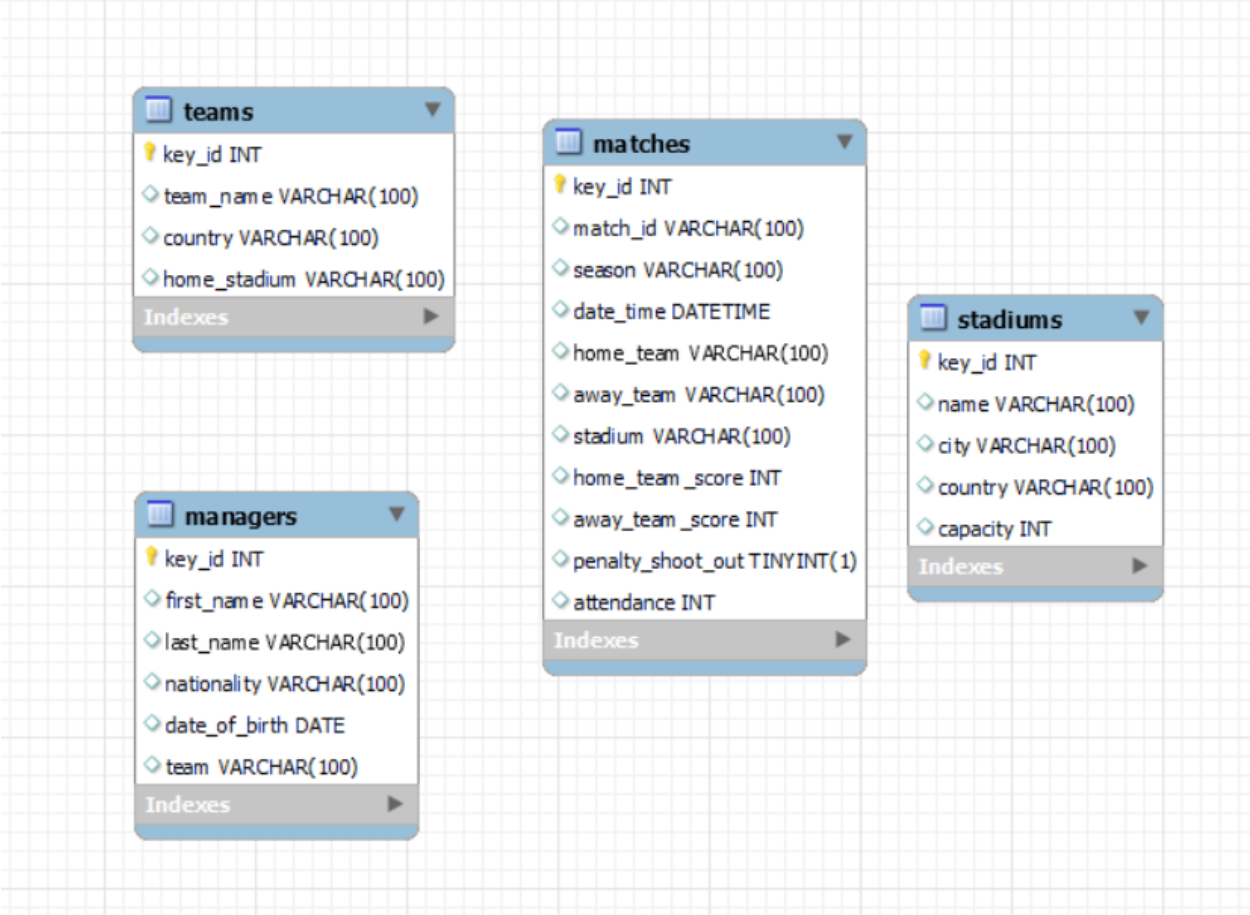
Код:

```
import csv
from datetime import datetime

def date_edit(row_name, source, dest, start_date_format, flag):
    with open(source, 'r') as csvfile:
        reader = csv.DictReader(csvfile)
        with open(dest, 'w', newline='') as outfile:
            fieldnames = reader.fieldnames
            writer = csv.DictWriter(outfile, fieldnames=fieldnames)
            writer.writeheader()
            for row in reader:
                date_str = row[row_name]
                date = datetime.strptime(date_str, start_date_format)
                updated_date_str = date.strftime('%Y-%m-%d')
                row[row_name] = updated_date_str
                if flag == 1:
                    old_value = row['match_id']
                    new_value = old_value[2:]
                    row['match_id'] = new_value
                writer.writerow(row)


if __name__ == "__main__":
    date_edit('date_of_birth', 'managers.csv', 'updated_managers.csv',
'%m/%d/%Y', 0)
    date_edit('date_time', 'matches.csv', 'updated_matches.csv', '%d-%b-%y
%I.%M.%S.%f000000 PM', 1)
```

```
"first_name","last_name","nationality","date_of_birth","team"
"Stefano","Pioli","Italy","10/19/1965","AC Milan"
"Alfred","Schreuder","Netherlands","11/2/1972","AFC Ajax"
"Gian Piero","Gasperini","Italy","1/26/1958","Atalanta"
"Diego","Simeone","Argentina","4/28/1970","Atlético Madrid"
"Julian","Nagelsmann","Germany","7/23/1987","Bayern München"
"Valérien","Ismaël","France","9/28/1975","Beşiktaş"
"Edin","Terzić","Germany","10/30/1982","Borussia Dortmund"
"Raphaël","Wicky","Switzerland","4/26/1977","BSC Young Boys"
```

```
first_name,last_name,nationality,date_of_birth,team
Stefano,Pioli,Italy,1965-10-19,AC Milan
Alfred,Schreuder,Netherlands,1972-11-02,AFC Ajax
Gian Piero,Gasperini,Italy,1958-01-26,Atalanta
Diego,Simeone,Argentina,1970-04-28,Atlético Madrid
Julian,Nagelsmann,Germany,1987-07-23,Bayern München
Valérien,Ismaël,France,1975-09-28,Beşiktaş
Edin,Terzić,Germany,1982-10-30,Borussia Dortmund
Raphaël,Wicky,Switzerland,1977-04-26,BSC Young Boys
```

Stage зона



Таблиця Stadiums містить:

- key_id - це автоматично збільшуване ціле число, яке використовується як первинний ключ.
- name, city і country - це рядки (VARCHAR), які можуть містити до 100 символів. Вони використовуються для зберігання назви стадіону, міста та країни відповідно.
- capacity - це ціле число (INT), яке використовується для зберігання місткості стадіону.

Ця таблиця дозволяє зберігати інформацію про стадіони, включаючи їх назву, місто та країну розташування, а також місткість. Це може бути корисним для аналізу даних про матчі, які проводилися на цих стадіонах, або

для вивчення впливу різних факторів (наприклад, місткості стадіону) на результати матчів.

Таблиця Teams містить:

- key_id - це автоматично збільшуване ціле число, яке використовується як первинний ключ.
- team_name, country і home_stadium - це рядки (VARCHAR), які можуть містити до 100 символів. Вони використовуються для зберігання назви команди, країни походження та домашнього стадіону відповідно.

Ця таблиця дозволяє зберігати інформацію про команди, включаючи їх назву, країну походження та домашній стадіон. Це може бути корисним для аналізу даних про матчі, які проводилися цими командами, або для вивчення впливу різних факторів (наприклад, домашнього стадіону) на результати матчів.

Таблиця Managers містить:

- key_id - це автоматично збільшуване ціле число, яке використовується як первинний ключ.
- first_name, last_name, nationality і team - це рядки (VARCHAR), які можуть містити до 100 символів. Вони використовуються для зберігання імені, прізвища, національності тренера та команди, яку він керує, відповідно.
- date_of_birth - це дата (DATE), яка використовується для зберігання дати народження тренера.

Ця таблиця дозволяє зберігати інформацію про тренерів, включаючи їх ім’я, національність, дату народження та команду, яку вони керують. Це може бути корисним для аналізу даних про тренерів та їх вплив на успіхи команд.

Завдяки первинному ключу key_id можна легко з’єднати дані з іншими таблицями, які містять пов’язану інформацію.

Таблиця Matches містить:

- key_id - це автоматично збільшуване ціле число, яке використовується як первинний ключ.
- match_id, season, home_team, away_team і stadium - це рядки (VARCHAR), які можуть містити до 100 символів. Вони використовуються для зберігання ідентифікатора матчу, сезону, назв домашньої та гостьової команд та стадіону відповідно.
- date_time - це дата та час (DATETIME), які використовуються для зберігання дати та часу проведення матчу.
- home_team_score, away_team_score і attendance - це цілі числа (INT), які використовуються для зберігання кількості голів, забитих домашньою та гостьовою командами, та кількості глядачів відповідно.
- penalty_shoot_out - це булеве значення (BOOLEAN), яке використовується для вказівки, чи була серія пенальті після матчу.

Ця таблиця дозволяє зберігати інформацію про матчі, включаючи ідентифікатор матчу, сезон, дату та час проведення, команди, стадіон, кількість голів, чи була серія пенальті та кількість глядачів. Це може бути корисним для аналізу даних про матчі та вивчення впливу різних факторів на результати матчів. Завдяки первинному ключу key_id можна легко з’єднати дані з іншими таблицями, які містять пов’язану інформацію.

Код створення Stage зони:

```
DROP database if exists STAGE;
CREATE database STAGE;
USE  STAGE;
CREATE TABLE Stadiums (
   key_id INT AUTO_INCREMENT,
```

```
    name VARCHAR(100),
    city VARCHAR(100),
    country VARCHAR(100),
    capacity INT,
    PRIMARY KEY (key_id)
);

CREATE TABLE Teams (
    key_id INT AUTO_INCREMENT,
    team_name VARCHAR(100),
    country VARCHAR(100),
    home_stadium VARCHAR(100),
    PRIMARY KEY (key_id)
);

CREATE TABLE Managers (
    key_id INT AUTO_INCREMENT,
    first_name VARCHAR(100),
    last_name VARCHAR(100),
    nationality VARCHAR(100),
    date_of_birth DATE,
    team VARCHAR(100),
    PRIMARY KEY (key_id)
);

CREATE TABLE Matches (
    key_id INT AUTO_INCREMENT,
    match_id VARCHAR(100),
    season VARCHAR(100),
    date_time DATETIME,
```

```
    home_team VARCHAR(100),
    away_team VARCHAR(100),
    stadium VARCHAR(100),
    home_team_score INT,
    away_team_score INT,
    penalty_shoot_out BOOLEAN,
    attendance INT,
    PRIMARY KEY (key_id)
);
```

Заповнення Stage зони

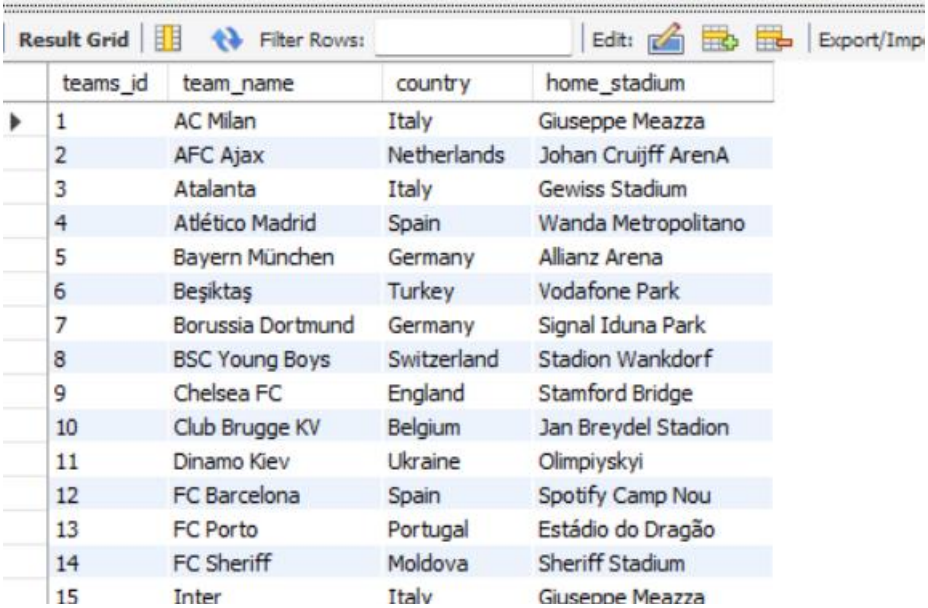
```
USE  STAGE;
SHOW VARIABLES LIKE 'secure_file_priv';

LOAD DATA INFILE 'C:/ProgramData/MySQL/MySQL Server
8.0/Uploads/stadiums.csv'
INTO TABLE Stadiums
FIELDS TERMINATED BY ','
ENCLOSED BY ""
LINES TERMINATED BY '\n'
IGNORE 1 ROWS
(name, city, country, capacity);


LOAD DATA INFILE 'C:/ProgramData/MySQL/MySQL Server
8.0/Uploads/teams.csv'
INTO TABLE Teams
FIELDS TERMINATED BY ','
ENCLOSED BY ""
LINES TERMINATED BY '\n'
IGNORE 1 ROWS
(team_name, country, home_stadium);


LOAD DATA INFILE 'C:/ProgramData/MySQL/MySQL Server
8.0/Uploads/managers.csv'
INTO TABLE Managers
FIELDS TERMINATED BY ','
ENCLOSED BY ""
LINES TERMINATED BY '\n'
IGNORE 1 ROWS
(first_name, last_name, nationality, date_of_birth, team);


LOAD DATA INFILE 'C:/ProgramData/MySQL/MySQL Server
8.0/Uploads/matches.csv'
INTO TABLE Matches
FIELDS TERMINATED BY ','
ENCLOSED BY ""
LINES TERMINATED BY '\n'
IGNORE 1 ROWS
(match_id, season, date_time, home_team, away_team, stadium,
home_team_score, away_team_score, penalty_shoot_out, attendance);
```

Таблиця Stadiums:

Result Grid | Filter Rows: | Edit: | Export/Impo

stadiums_id	name	city	country	capacity
1	Giuseppe Meazza	Milano	Italy	75923
2	Johan Cruijff ArenA	Amsterdam	Netherlands	54990
3	Gewiss Stadium	Bergamo	Italy	26562
4	Wanda Metropolitano	Madrid	Spain	68000
5	Allianz Arena	München	Germany	75024
6	Vodafone Park	Istanbul	Turkey	41903
7	Signal Iduna Park	Dortmund	Germany	81365
8	Stadion Wankdorf	Bern	Switzerland	32000
9	Stamford Bridge	London	England	41837
10	Jan Breydel Stadion	Brugge	Belgium	29042
11	Olimpiyskyi	Kiev	Ukraine	70050
12	Spotify Camp Nou	Barcelona	Spain	99354
13	Estádio do Dragão	Porto	Portugal	54378
14	Sheriff Stadium	Tiraspol	Moldova	14300
15	Allianz Stadium	Torino	Italy	41254
16	Stade Pierre Mauroy	Villeneuve ...	France	50186
17	Anfield	Liverpool	England	54074

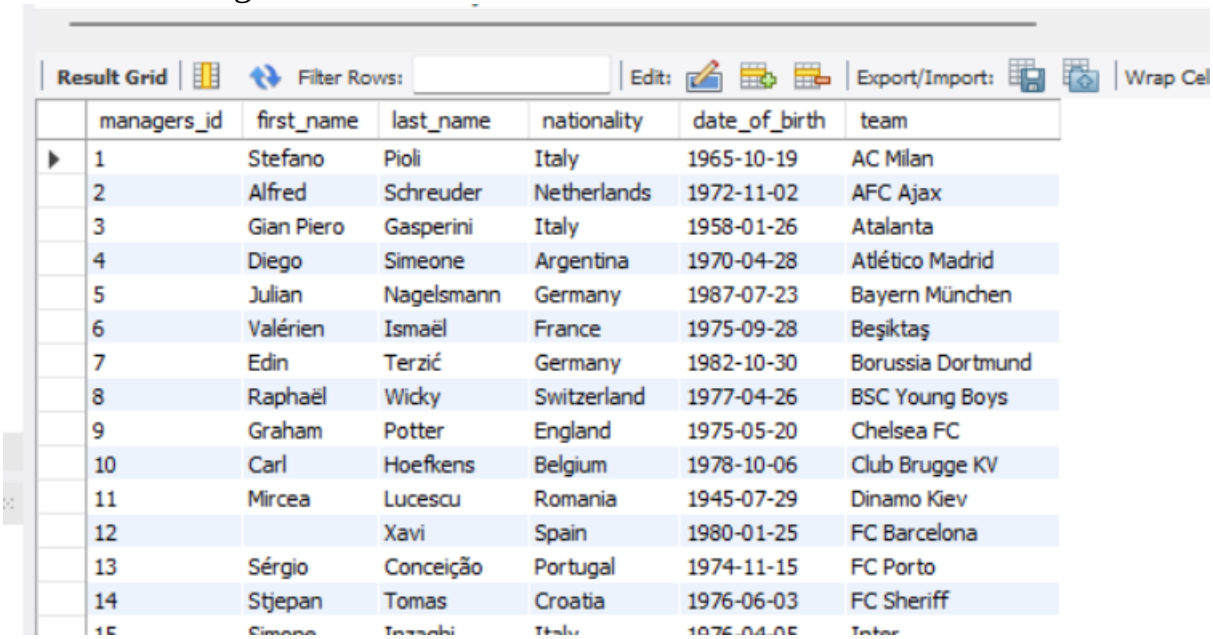
Таблиця Teams:



Result Grid | Filter Rows: | Edit: | Export/Imp

teams_id	team_name	country	home_stadium
1	AC Milan	Italy	Giuseppe Meazza
2	AFC Ajax	Netherlands	Johan Cruijff ArenA
3	Atalanta	Italy	Gewiss Stadium
4	Atlético Madrid	Spain	Wanda Metropolitano
5	Bayern München	Germany	Allianz Arena
6	Beşiktaş	Turkey	Vodafone Park
7	Borussia Dortmund	Germany	Signal Iduna Park
8	BSC Young Boys	Switzerland	Stadion Wankdorf
9	Chelsea FC	England	Stamford Bridge
10	Club Brugge KV	Belgium	Jan Breydel Stadion
11	Dinamo Kiev	Ukraine	Olimpiyskyi
12	FC Barcelona	Spain	Spotify Camp Nou
13	FC Porto	Portugal	Estádio do Dragão
14	FC Sheriff	Moldova	Sheriff Stadium
15	Inter	Italy	Giuseppe Meazza

Таблиця Managers:



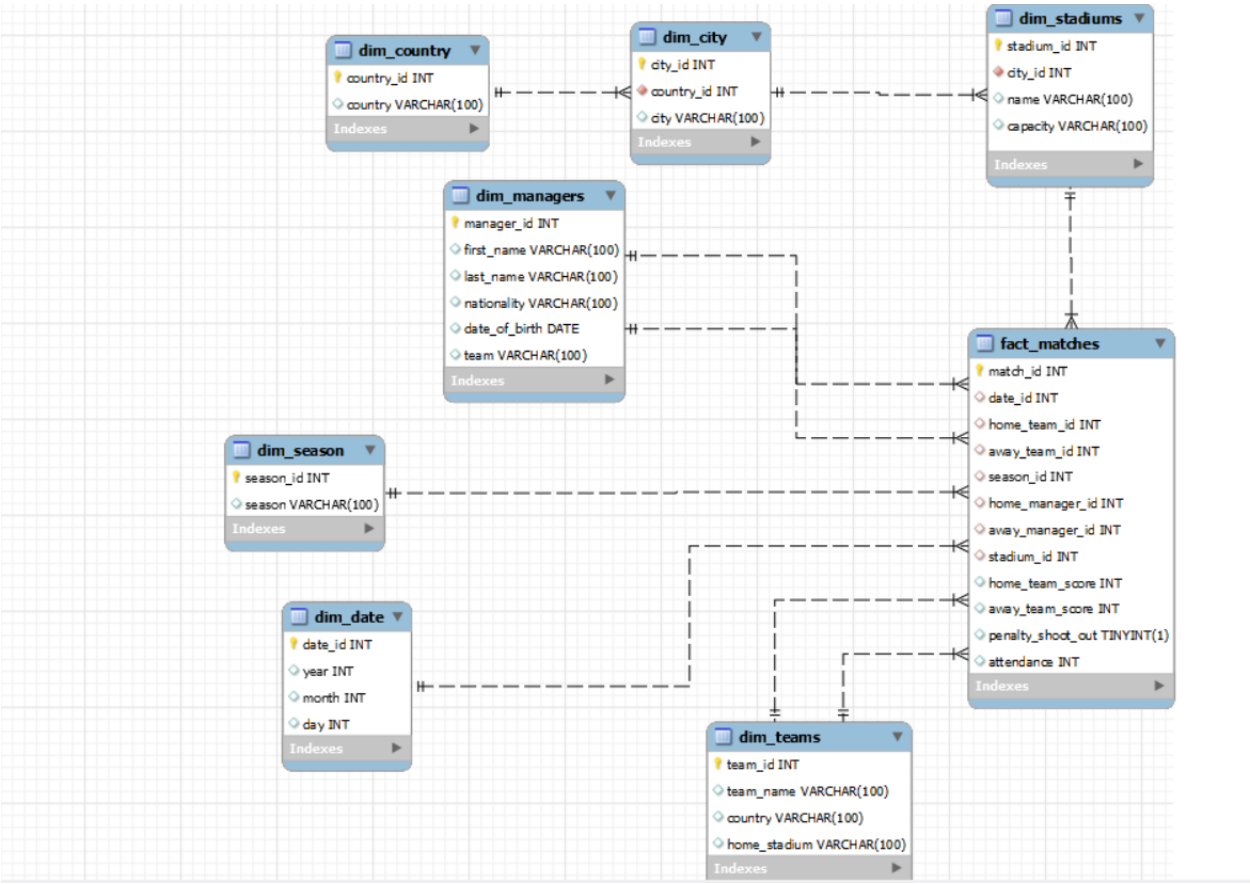
managers_id	first_name	last_name	nationality	date_of_birth	team
1	Stefano	Pioli	Italy	1965-10-19	AC Milan
2	Alfred	Schreuder	Netherlands	1972-11-02	AFC Ajax
3	Gian Piero	Gasperini	Italy	1958-01-26	Atalanta
4	Diego	Simeone	Argentina	1970-04-28	Atlético Madrid
5	Julian	Nagelsmann	Germany	1987-07-23	Bayern München
6	Valérien	Ismaël	France	1975-09-28	Beşiktaş
7	Edin	Terzić	Germany	1982-10-30	Borussia Dortmund
8	Raphaël	Wicky	Switzerland	1977-04-26	BSC Young Boys
9	Graham	Potter	England	1975-05-20	Chelsea FC
10	Carl	Hoefkens	Belgium	1978-10-06	Club Brugge KV
11	Mircea	Lucescu	Romania	1945-07-29	Dinamo Kiev
12		Xavi	Spain	1980-01-25	FC Barcelona
13	Sérgio	Conceição	Portugal	1974-11-15	FC Porto
14	Stjepan	Tomas	Croatia	1976-06-03	FC Sheriff

Таблиця Matches:

match_id	season	date_time	home_team	away_team	stadium	home_team_score	away_team_score	penalty_shoot_out	attendanc
1	2021-2022	2021-09-15 00:00:00	Manchester City	RB Leipzig	Etihad Stadium	6	3	0	38062
2	2021-2022	2021-09-15 00:00:00	Club Brugge KV	Paris Saint-Germain	Jan Breydel Stadion	1	1	0	27546
3	2021-2022	2021-09-28 00:00:00	Paris Saint-Germain	Manchester City	Parc des Princes	2	0	0	37350
4	2021-2022	2021-09-28 00:00:00	RB Leipzig	Club Brugge KV	Red Bull Arena	1	2	0	23500
5	2021-2022	2021-10-19 00:00:00	Club Brugge KV	Manchester City	Jan Breydel Stadion	1	5	0	24915
6	2021-2022	2021-10-19 00:00:00	Paris Saint-Germain	RB Leipzig	Parc des Princes	3	2	0	47359
7	2021-2022	2021-11-03 00:00:00	RB Leipzig	Paris Saint-Germain	Red Bull Arena	2	2	0	39794
8	2021-2022	2021-11-03 00:00:00	Manchester City	Club Brugge KV	Etihad Stadium	4	1	0	50228
9	2021-2022	2021-11-24 00:00:00	Manchester City	Paris Saint-Germain	Etihad Stadium	2	1	0	52030
10	2021-2022	2021-11-24 00:00:00	Club Brugge KV	RB Leipzig	Jan Breydel Stadion	0	5	0	24072
11	2021-2022	2021-12-07 00:00:00	RB Leipzig	Manchester City	Red Bull Arena	2	1	0	0
12	2021-2022	2021-12-07 00:00:00	Paris Saint-Germain	Club Brugge KV	Parc des Princes	4	1	0	47492
13	2021-2022	2021-09-15 00:00:00	Atlético Madrid	FC Porto	Wanda Metropolitano	0	0	0	40098
14	2021-2022	2021-09-15 00:00:00	Liverpool FC	AC Milan	Anfield	3	2	0	51445

Отже, дані успішно завантажені до Stage зони

Модель основного сховища



У моїй базі даних я маю фактову таблицю, яка називається fact_Matches. Ця таблиця є центральною частиною схеми зірки і містить інформацію про кожен окремий матч, який відбувся.

Кожен запис в таблиці fact_Matches відповідає одному матчу. Він містить інформацію про дату матчу, команди, які брали участь, сезон, в якому він відбувся, тренерів обох команд, стадіон, на якому він відбувся, а також результати матчу.

Результати матчу включають кількість голів, забитих кожною командою, чи була серія пенальті, та відвідуванність матчу. Ці дані дозволяють нам аналізувати результати матчів та вивчати різні аспекти гри.

Таблиця fact_Matches містить зовнішні ключі на таблиці виміри, які містять детальні дані про команди, стадіони, тренерів, сезони та дати. Ці таблиці виміри надають мені додаткову інформацію, яка допомагає краще розуміти контекст кожного матчу.

Наприклад, таблиця dim_Teams містить інформацію про кожну команду, включаючи її назву, країну та домашній стадіон. Таблиця dim_Managers містить інформацію про тренерів, включаючи їх імена, національність, дату народження та команду, яку вони тренують. Таблиці dim__Season та dim_Date містять інформацію про сезони та дати відповідно.

Використовуючи цю схему зірки, я можу легко виконувати складні запити до бази даних та отримувати цінну інформацію про матчі, команди, тренерів та інші аспекти нашого дослідження.

Код створення основного сховища:

```
DROP database MAIN_STORAGE;
CREATE database MAIN_STORAGE;
USE MAIN_STORAGE;


CREATE TABLE dim_Teams (
    team_id INT AUTO_INCREMENT,
    team_name VARCHAR(100),
    country VARCHAR(100),
    home_stadium VARCHAR(100),
    PRIMARY KEY (team_id)
);

CREATE TABLE dim_Date (
    date_id INT AUTO_INCREMENT,
    year INT,
    month INT,
    day INT,
    PRIMARY KEY (date_id)
);

CREATE TABLE dim_Season (
    season_id INT AUTO_INCREMENT,
    season VARCHAR(100),
    PRIMARY KEY (season_id)
);

CREATE TABLE dim_Country (
    country_id INT AUTO_INCREMENT,
    country VARCHAR(100),
    PRIMARY KEY (country_id)
);

CREATE TABLE dim_City (
    city_id INT AUTO_INCREMENT,
    country_id INT NOT NULL,
    city VARCHAR(100),
    PRIMARY KEY (city_id),
    FOREIGN KEY (country_id) REFERENCES dim_Country (country_id)
);

CREATE TABLE dim_Stadium (
    stadium_id INT AUTO_INCREMENT,
```

```
    source_id INT DEFAULT NULL,
    city_id INT NOT NULL,
    name VARCHAR(100),
    capacity VARCHAR(100),
    start_date DATE DEFAULT NULL,
    end_date DATE DEFAULT NULL,
    PRIMARY KEY (stadium_id),
    FOREIGN KEY (city_id) REFERENCES dim_City (city_id)
);

CREATE TABLE dim_Managers (
    manager_id INT AUTO_INCREMENT,
    first_name VARCHAR(100),
    last_name VARCHAR(100),
    nationality VARCHAR(100),
    date_of_birth DATE,
    team VARCHAR(100),
    PRIMARY KEY (manager_id)
);

CREATE TABLE fact_Matches (
    match_id INT AUTO_INCREMENT,
    date_id INT,
    home_team_id INT,
    away_team_id INT,
    season_id INT,
    home_manager_id INT,
    away_manager_id INT,
    stadium_id INT,
    home_team_score INT,
    away_team_score INT,
    penalty_shoot_out BOOLEAN,
    attendance INT,
    PRIMARY KEY (match_id),
    FOREIGN KEY (date_id) REFERENCES dim_Date (date_id),
    FOREIGN KEY (home_team_id) REFERENCES dim_Teams (team_id),
    FOREIGN KEY (away_team_id) REFERENCES dim_Teams (team_id),
    FOREIGN KEY (season_id) REFERENCES dim_Season (season_id),
    FOREIGN KEY (home_manager_id) REFERENCES dim_Managers
(manager_id),
    FOREIGN KEY (away_manager_id) REFERENCES dim_Managers
(manager_id),
    FOREIGN KEY (stadium_id) REFERENCES dim_Stadium (stadium_id)
);
```

Заповнення основного сховища

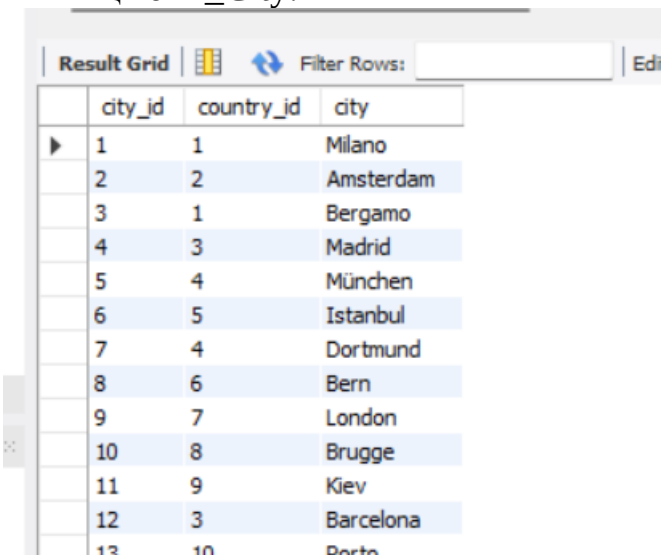
```
USE STAGE;

INSERT INTO MAIN_STORAGE.dim_Date(year, month, day)
SELECT DISTINCT YEAR(date_time), MONTH(date_time), DAY(date_time)
FROM matches;

INSERT INTO MAIN_STORAGE.dim_City(country_id, city)
SELECT DISTINCT dc.country_id, city
FROM stadiums
JOIN MAIN_STORAGE.dim_Country dc ON dc.country = stadiums.country;

INSERT INTO MAIN_STORAGE.dim_Season(season)
SELECT DISTINCT season
FROM matches;

INSERT INTO MAIN_STORAGE.dim_Teams(team_name, country,
home_stadium)
SELECT DISTINCT team_name, country, home_stadium
FROM teams;

INSERT INTO MAIN_STORAGE.dim_Managers(first_name, last_name,
nationality, date_of_birth, team)
SELECT DISTINCT first_name, last_name, nationality, date_of_birth, team
FROM managers;

INSERT INTO MAIN_STORAGE.dim_Country(country)
SELECT DISTINCT country
FROM stadiums;

INSERT INTO MAIN_STORAGE.dim_Stadiums(city_id, name, capacity)
SELECT DISTINCT city_id, name, capacity
FROM stadiums
JOIN MAIN_STORAGE.dim_City ds ON ds.city = stadiums.city;

-- Вставляємо date_id
INSERT INTO MAIN_STORAGE.fact_Matches(date_id)
SELECT dd.date_id
FROM STAGE.Matches m
JOIN MAIN_STORAGE.dim_Date dd ON YEAR(m.date_time) = dd.year AND
MONTH(m.date_time) = dd.month AND DAY(m.date_time) = dd.day;

-- Вставляємо home_team_id
UPDATE MAIN_STORAGE.fact_Matches fm
```

```
JOIN STAGE.Matches m ON fm.match_id = m.match_id
JOIN MAIN_STORAGE.dim_Teams dt ON m.home_team = dt.team_name
SET fm.home_team_id = dt.team_id;

-- Вставляємо away_team_id
UPDATE MAIN_STORAGE.fact_Matches fm
JOIN STAGE.Matches m ON fm.match_id = m.match_id
JOIN MAIN_STORAGE.dim_Teams dt ON m.away_team = dt.team_name
SET fm.away_team_id = dt.team_id;

-- Вставляємо season_id
UPDATE MAIN_STORAGE.fact_Matches fm
JOIN STAGE.Matches m ON fm.match_id = m.match_id
JOIN MAIN_STORAGE.dim_Season ds ON m.season = ds.season
SET fm.season_id = ds.season_id;

-- Вставляємо stadium_id
UPDATE MAIN_STORAGE.fact_Matches fm
JOIN STAGE.Matches m ON fm.match_id = m.match_id
JOIN MAIN_STORAGE.dim_Stadiums dst ON m.stadium = dst.name
SET fm.stadium_id = dst.stadium_id;

-- Вставляємо home_team_score
UPDATE MAIN_STORAGE.fact_Matches fm
JOIN STAGE.Matches m ON fm.match_id = m.match_id
SET fm.home_team_score = m.home_team_score;

-- Вставляємо away_team_score
UPDATE MAIN_STORAGE.fact_Matches fm
JOIN STAGE.Matches m ON fm.match_id = m.match_id
SET fm.away_team_score = m.away_team_score;

SET SQL_SAFE_UPDATES = 0;


-- Вставляємо penalty_shoot_out
UPDATE MAIN_STORAGE.fact_Matches fm
JOIN STAGE.Matches m ON fm.match_id = m.match_id
SET fm.penalty_shoot_out = m.penalty_shoot_out;

-- Вставляємо attendance
UPDATE MAIN_STORAGE.fact_Matches fm
JOIN STAGE.Matches m ON fm.match_id = m.match_id
SET fm.attendance = m.attendance;
```

Таблиця dim_Teams:

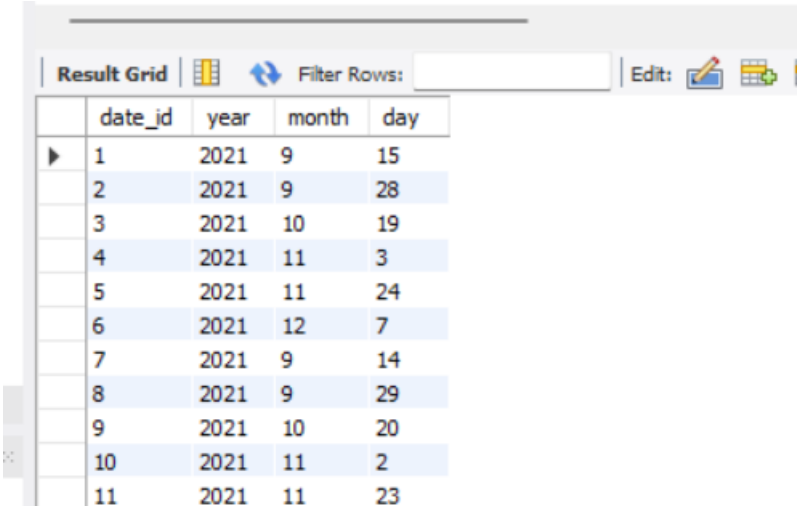
team_id	team_name	country	home_stadium
1	AC Milan	Italy	Giuseppe Meazza
2	AFC Ajax	Netherlands	Johan Cruijff ArenA
3	Atalanta	Italy	Gewiss Stadium
4	Atlético Madrid	Spain	Wanda Metropolitano
5	Bayern München	Germany	Allianz Arena
6	Beşiktaş	Turkey	Vodafone Park
7	Borussia Dortmund	Germany	Signal Iduna Park
8	BSC Young Boys	Switzerland	Stadion Wankdorf
9	Chelsea FC	England	Stamford Bridge
10	Club Brugge KV	Belgium	Jan Breydel Stadion

Таблиця dim_City:



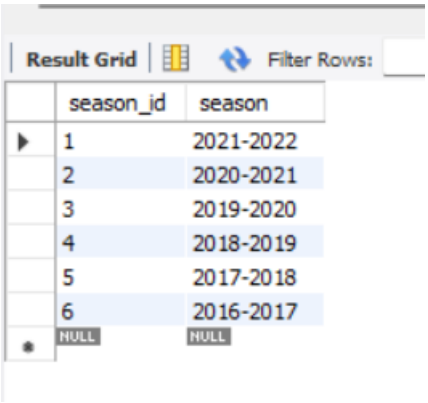
city_id	country_id	city
1	1	Milano
2	2	Amsterdam
3	1	Bergamo
4	3	Madrid
5	4	München
6	5	Istanbul
7	4	Dortmund
8	6	Bern
9	7	London
10	8	Brugge
11	9	Kiev
12	3	Barcelona

Таблиця dim_Date:



date_id	year	month	day
1	2021	9	15
2	2021	9	28
3	2021	10	19
4	2021	11	3
5	2021	11	24
6	2021	12	7
7	2021	9	14
8	2021	9	29
9	2021	10	20
10	2021	11	2
11	2021	11	23

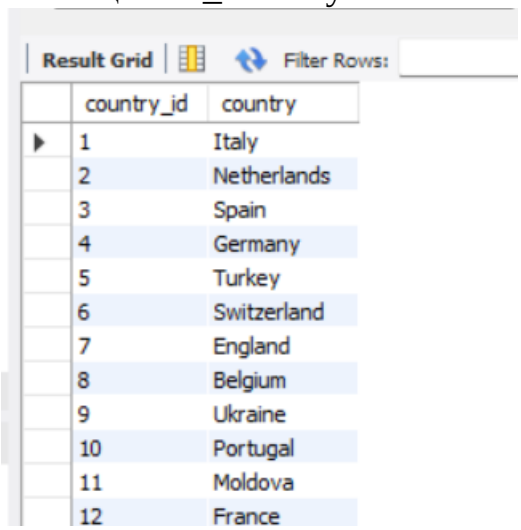
Таблиця dim_Season:



Result Grid | Filter Rows:

season_id	season
1	2021-2022
2	2020-2021
3	2019-2020
4	2018-2019
5	2017-2018
6	2016-2017
NULL	NULL

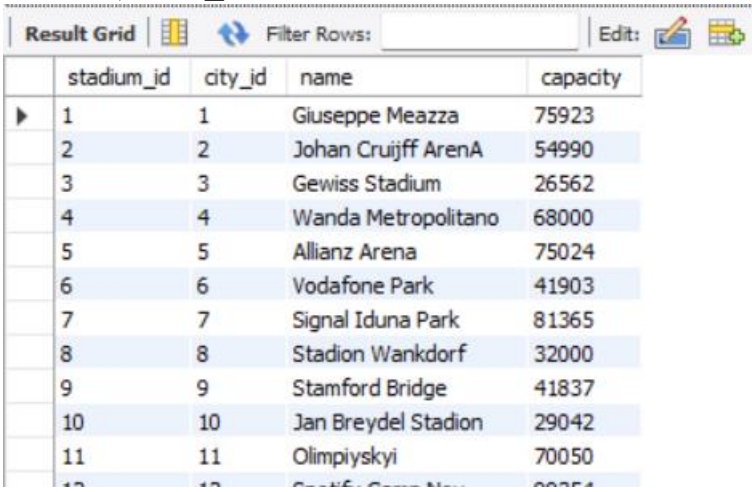
Таблиця dim_Country:



Result Grid | Filter Rows:

country_id	country
1	Italy
2	Netherlands
3	Spain
4	Germany
5	Turkey
6	Switzerland
7	England
8	Belgium
9	Ukraine
10	Portugal
11	Moldova
12	France

Таблиця dim_Stadiums:



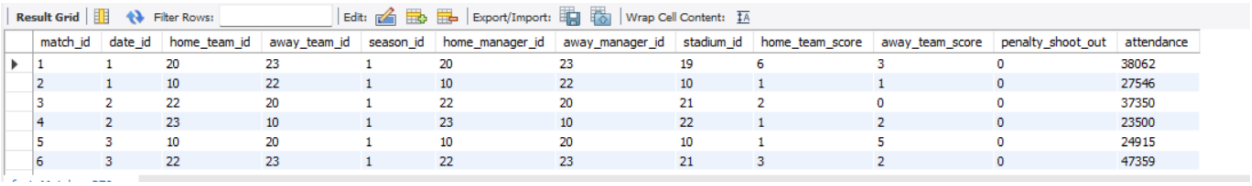
Result Grid | Filter Rows: | Edit:

stadium_id	city_id	name	capacity
1	1	Giuseppe Meazza	75923
2	2	Johan Cruijff ArenA	54990
3	3	Gewiss Stadium	26562
4	4	Wanda Metropolitano	68000
5	5	Allianz Arena	75024
6	6	Vodafone Park	41903
7	7	Signal Iduna Park	81365
8	8	Stadion Wankdorf	32000
9	9	Stamford Bridge	41837
10	10	Jan Breydel Stadion	29042
11	11	Olimpiyskyi	70050

Таблиця dim_Managers:

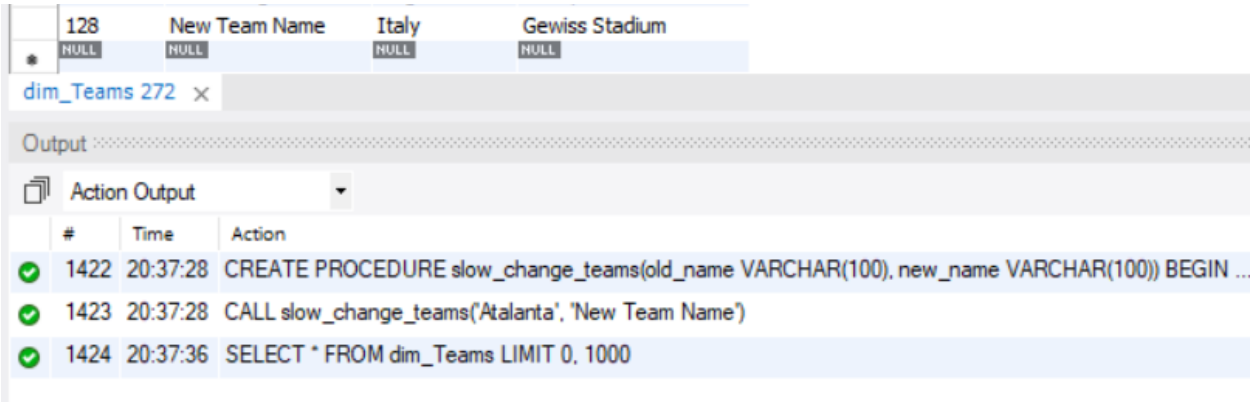
manager_id	first_name	last_name	nationality	date_of_birth	team
1	Stefano	Pioli	Italy	1965-10-19	AC Milan
2	Alfred	Schreuder	Netherlands	1972-11-02	AFC Ajax
3	Gian Piero	Gasperini	Italy	1958-01-26	Atalanta
4	Diego	Simeone	Argentina	1970-04-28	Atlético Madrid
5	Julian	Nagelsmann	Germany	1987-07-23	Bayern München
6	Valérien	Ismaël	France	1975-09-28	Beşiktaş
7	Edin	Terzić	Germany	1982-10-30	Borussia Dortmund
8	Raphaël	Wicky	Switzerland	1977-04-26	BSC Young Boys
9	Graham	Potter	England	1975-05-20	Chelsea FC
10	Carl	Hoefkens	Belgium	1978-10-06	Club Brugge KV
11	Mircea	Lucescu	Romania	1945-07-29	Dinamo Kiev
12		Xavi	Spain	1980-01-25	FC Barcelona
13	Sérgio	Conceição	Portugal	1974-11-15	FC Porto

Таблиця fact_Matches:



match_id	date_id	home_team_id	away_team_id	season_id	home_manager_id	away_manager_id	stadium_id	home_team_score	away_team_score	penalty_shoot_out	attendance
1	1	20	23	1	20	23	19	6	3	0	38062
2	1	10	22	1	10	22	10	1	1	0	27546
3	2	22	20	1	22	20	21	2	0	0	37350
4	2	23	10	1	23	10	22	1	2	0	23500
5	3	10	20	1	10	20	10	1	5	0	24915
6	3	22	23	1	22	23	21	3	2	0	47359

slowly changing dimension



Моя база даних містить інформацію про команди, менеджерів, стадіони тощо. Ці сутності можуть змінюватися з часом. Наприклад, команда може змінити свою назву, менеджер може перейти до іншої команди, а стадіон може змінити свою ємність.

SCD дозволяє відслідковувати ці зміни в часі. Тому мій код використовує SCD для відслідковування змін в назвах команд. Коли команда змінює свою назву, я створюю новий запис в таблиці dim_Teams з новою назвою, але зберігаю також і старі значення для країни та домашнього стадіону. Також оновлюю відповідні записи в таблиці fact_Matches, щоб вони відображали нову назву команди. Це дозволяє відслідковувати, як зміна назви команди впливає на її виступи в матчах.

Використання SCD допомагає забезпечити точність та цілісність даних. Воно також дозволяє зберігати більш детальну історію змін, що може бути корисною для глибокого аналізу та прогнозування.

Цей код створює процедуру , яка змінює назву команди в таблиці dim_Teams та оновлює відповідні записи в таблиці fact_Matches:

```
USE MAIN_STORAGE;
DROP PROCEDURE IF EXISTS slow_change_teams;
DELIMITER //
CREATE PROCEDURE slow_change_teams(old_name VARCHAR(100),
new_name VARCHAR(100))
BEGIN
 DECLARE old_id INT DEFAULT NULL;
 DECLARE old_country VARCHAR(100);
 DECLARE old_home_stadium VARCHAR(100);

 SELECT team_id, country, home_stadium
 INTO old_id, old_country, old_home_stadium
 FROM dim_Teams
```

```
 WHERE team_name = old_name;

 IF old_id IS NULL THEN
  SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = 'The old name of the
team does not exist';
 ELSE
  INSERT INTO dim_Teams (team_name, country, home_stadium)
  VALUES (new_name, old_country, old_home_stadium);

  UPDATE fact_Matches
  SET home_team_id = (SELECT team_id FROM dim_Teams WHERE
team_name = new_name)
  WHERE home_team_id = old_id;

  UPDATE fact_Matches
  SET away_team_id = (SELECT team_id FROM dim_Teams WHERE
team_name = new_name)
  WHERE away_team_id = old_id;
 END IF;
END //
DELIMITER ;


CALL slow_change_teams('Atalanta', 'New Team Name');
```

Incremental load

Поступове завантаження відіграє важливу роль у забезпеченні актуальності вашого сховища даних, мінімізуючи при цьому використання ресурсів. Воно передбачає ідентифікацію та вибіркове завантаження лише нових записів, що оптимізує продуктивність та зменшує накладні витрати на обробку.

Моя база даних, яка включає таблиці команд, менеджерів, стадіонів та матчів, використовує стратегію поступового завантаження для ефективного оновлення даних. Це зроблено шляхом ідентифікації нових записів в проміжній області та їх вибіркового завантаження в відповідні таблиці в сховищі.

Код:

```
USE STAGE;
-- Завантажте нові команди в таблицю dim_Teams
INSERT INTO MAIN_STORAGE.dim_Teams (team_name, country, home_stadium)
SELECT DISTINCT team_name, country, home_stadium
FROM STAGE.Teams t
WHERE NOT EXISTS (
SELECT 1
FROM MAIN_STORAGE.dim_Teams dt
WHERE dt.team_name = t.team_name
);

-- Завантажте нові стадіони в таблицю dim_Stadiums
INSERT INTO MAIN_STORAGE.dim_Stadiums (name, city_id, capacity)
SELECT DISTINCT s.name, c.city_id, s.capacity
FROM STAGE.Stadiums s
JOIN MAIN_STORAGE.dim_City c ON s.city = c.city
WHERE NOT EXISTS (
SELECT 1
FROM MAIN_STORAGE.dim_Stadiums ds
WHERE ds.name = s.name
);

-- Завантажте нових менеджерів в таблицю dim_Managers
INSERT INTO MAIN_STORAGE.dim_Managers (first_name, last_name, nationality, date_of_birth, team)
SELECT DISTINCT first_name, last_name, nationality, date_of_birth, team
FROM STAGE.Managers m
WHERE NOT EXISTS (
SELECT 1
FROM MAIN_STORAGE.dim_Managers dm
```

```
WHERE dm.first_name = m.first_name AND dm.last_name = m.last_name
);

-- Завантажте нові матчі в таблицю fact_Matches
INSERT INTO MAIN_STORAGE.fact_Matches (date_id, home_team_id,
away_team_id, season_id, home_manager_id, away_manager_id, stadium_id,
home_team_score, away_team_score, penalty_shoot_out, attendance)
SELECT d.date_id, ht.team_id, at.team_id, s.season_id, hm.manager_id,
am.manager_id, st.stadium_id, m.home_team_score, m.away_team_score,
m.penalty_shoot_out, m.attendance
FROM STAGE.Matches m
JOIN MAIN_STORAGE.dim_Date d ON DATE(m.date_time) =
CONCAT(d.year, '-', d.month, '-', d.day)
JOIN MAIN_STORAGE.dim_Teams ht ON m.home_team = ht.team_name
JOIN MAIN_STORAGE.dim_Teams at ON m.away_team = at.team_name
JOIN MAIN_STORAGE.dim_Season s ON m.season = s.season
JOIN MAIN_STORAGE.dim_Managers hm ON ht.team_name = hm.team
JOIN MAIN_STORAGE.dim_Managers am ON at.team_name = am.team
JOIN MAIN_STORAGE.dim_Stadiums st ON m.stadium = st.name
WHERE NOT EXISTS (
SELECT 1
FROM MAIN_STORAGE.fact_Matches fm
WHERE fm.match_id = m.match_id
);
```

Висновок

У цьому проекті було розроблено сховище даних, спеціально призначене для зберігання інформації, пов’язаної з футболом. Основний акцент було зроблено на впровадженні стратегії поступового завантаження, що гарантує, що лише нові або змінені дані з проміжної бази даних (STAGE) вставляються в відповідні таблиці сховища (MAIN_STORAGE). Цей підхід мінімізує використання ресурсів і забезпечує консистентність даних.

База даних, яка включає таблиці `dim_Teams`, `dim_Managers`, `dim_Stadiums` та `fact_Matches`, використовує цю стратегію для ефективного оновлення даних. Це забезпечує, що сховище завжди містить найновішу інформацію, зберігаючи при цьому цілісність і послідовність даних.

Цей підхід до поступового завантаження даних дозволяє максимально ефективно використовувати ресурси, забезпечуючи при цьому актуальність даних у сховищі.