*Додаток 1*

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

Звіт

з комп’ютерного практикуму № 1 з дисципліни

«Аналіз даних в інформаційних системах»

на тему: «Створення сховища даних»

Виконав студент ІП-13, Бондаренко Максим Вікторович

(шифр, прізвище, ім'я, по батькові)

Перевірив Олійник Юрій Олександрович

( прізвище, ім'я, по батькові)

Київ 202 3

Комп’ютерний практикум 1

Тема – створення сховища даних.

Мета – ознайомитись з підходами до створення сховищ даних.

Для виконання даної лабораторної роботи було вибрано даний набір даних: <https://www.kaggle.com/datasets/cbxkgl/uefa-champions-league-2016-2022-data>

Даний набір містить дані про всі матчі Ліги Чемпіонів УЄФА в період з 2016 по 2022 роки. Для обробки візьмемо чотири початкових таблиці: managers, stadiums, teams та matches.

Таблиця managers містить дані про всіх тренерів: їх національність, дата народження та команда, яку вони тренують.

Таблиця stadiums містить інгформацію про всі стадіони: їх назва, місто та країна знаходження, а також місткість.

Таблиця teams містить інформацію про всі команди, що приймали участь у турнірі: їх назва, країна походження та домашній стадіон.

Таблиця matches мість дані про всі матчі цього періоду: сезон, дата та час, назви домашньої та гостьової команди, стадіон, кількість голів домашньої та виїздної команди, чи була серія пенальті та відвідування матчу глядачами.

Форматування даних

Для форматування даних використаємо python скрипт наведений нижче:

main.py

from funcs import \*  
  
if \_\_name\_\_ == "\_\_main\_\_":  
 format\_date('date\_of\_birth', 'managers.csv', 'updated\_managers.csv', '%m-%d-%Y', 0)  
 format\_date('date\_time', 'matches.csv', 'updated\_matches.csv', '%d-%b-%y %I.%M.%S.%f000000 PM', 1)  
 format\_stadiums\_names('updated\_matches.csv')

funcs.py

import csv  
from datetime import datetime  
def format\_date(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)  
  
def format\_stadiums\_names(source):  
 with open(source, 'r') as csvfile:  
 reader = csv.DictReader(csvfile)  
 rows = list(reader)  
  
 for row in rows:  
 if row['home\_team'] == 'RB Leipzig' and row['stadium'] == 'Red Bull Arena':  
 row['stadium'] = 'Red Bull Arena (Leipzig)'  
 elif row['home\_team'] == 'RB Salzburg' and row['stadium'] == 'Red Bull Arena':  
 row['stadium'] = 'Red Bull Arena (Salzburg)'  
  
 with open(source, 'w', newline='') as csvfile:  
 fieldnames = ['match\_id', 'season', 'date\_time', 'home\_team', 'away\_team', 'stadium', 'home\_team\_score',  
 'away\_team\_score', 'isPenalty', 'attendance']  
 writer = csv.DictWriter(csvfile, fieldnames=fieldnames)  
  
 writer.writeheader()  
 for row in rows:  
 writer.writerow(row)

Stage зона

Graphical user interface

Description automatically generated with medium confidence

**Matches**

season – поле для зберігання сезону.

date\_time – поле для зберігання дати матчу.

home\_team – команда, що виступає вдома.

away\_team – команда, що виступає в гостях.

stadium – назва стадіону, де відбувався матч.

home\_team\_score – кількість голів, забитих домашньою командою.

away\_team\_score – кількість голів, забитих гостьовою командою.

penalty\_shoot\_out – чи відбувалась серія пенальті в матчі.

attendance – відвідуванність матчу глядачами.

**Stadiums**

name – назва стадіону.

city – місто знаходження стадіону.

country – країна знаходження стадіону.

capacity – місткість стадіону.

**Managers**

first\_name – ім’я тренера.

last\_name – прізвище тренера.

nationality – національність.

date\_of\_birth – дата народження.

team – команда, яку тренує.

**Teams**

team\_name – назва команди.

country – країна походження.

home\_stadium – домашнйі стадіон.

Для створення stage зони було задіяно sql скрипт нижче:

drop  
 database if exists stage;  
create  
 database stage;  
  
use  
 stage;  
  
drop table if exists matches;  
drop table if exists stadiums;  
drop table if exists teams;  
  
create table matches  
(  
 id int not null,  
 season varchar(9) not null,  
 date\_time datetime not null,  
 home\_team varchar(50) not null,  
 away\_team varchar(50) not null,  
 stadium varchar(50) not null,  
 home\_team\_score int not null,  
 away\_team\_score int not null,  
 penalty\_shoot\_out tinyint(1) not null,  
 attendance int not null,  
 primary key (id)  
);  
  
create table stadiums  
(  
 id int not null,  
 name varchar(50) not null,  
 city varchar(50) not null,  
 country varchar(50) not null,  
 capacity int not null,  
 primary key (id)  
);  
  
create table teams  
(  
 id int not null,  
 team\_name varchar(50),  
 country varchar(50),  
 home\_stadium varchar(50),  
 primary key (id)  
);  
  
create table managers  
(  
 id int not null,  
 first\_name varchar(50) not null,  
 last\_name varchar(50) not null,  
 nationality varchar(40) not null,  
 date\_of\_birth date not null,  
 team varchar(50),  
 primary key (id)  
);

Скрипт для заповнення stage зони даними наведено нижче:

use stage;  
  
truncate table matches;  
truncate table stadiums;  
truncate table teams;  
truncate table managers;  
  
LOAD DATA INFILE 'C:/ProgramData/MySQL/MySQL Server 8.0/Uploads/matches.csv' INTO TABLE matches  
 FIELDS TERMINATED BY ','  
 LINES TERMINATED BY '\n'  
 IGNORE 1 ROWS;  
  
LOAD DATA INFILE 'C:/ProgramData/MySQL/MySQL Server 8.0/Uploads/stadiums.csv' INTO TABLE stadiums  
 FIELDS TERMINATED BY ','  
 LINES TERMINATED BY '\n'  
 IGNORE 1 ROWS;  
  
LOAD DATA INFILE 'C:/ProgramData/MySQL/MySQL Server 8.0/Uploads/teams.csv' INTO TABLE teams  
 FIELDS TERMINATED BY ','  
 LINES TERMINATED BY '\n'  
 IGNORE 1 ROWS;  
  
LOAD DATA INFILE 'C:/ProgramData/MySQL/MySQL Server 8.0/Uploads/managers.csv' INTO TABLE managers  
 FIELDS TERMINATED BY ','  
 LINES TERMINATED BY '\n'  
 IGNORE 1 ROWS;

Main сховище

A screenshot of a computer

Description automatically generated with low confidence

Фактова таблиця містить зовнішні ключі на таблиці виміри, що містять детальні дані про команди, стадіони, тренерів, сезони та дати. Також фактова таблиця містить інформацію про кількість забитих голів кожною командою, серію пенальті та відвідуванність матчу.

Скрипти для створення основного сховища наведені нижче:

drop database if exists main\_warehouse;  
create database main\_warehouse;  
  
use main\_warehouse;  
  
drop table if exists dimStadium;  
drop table if exists dimLocation;  
drop table if exists dimManager;  
drop table if exists dimSeason;  
drop table if exists dimDate;  
drop table if exists dimTeams;  
drop table if exists factMatches;  
  
create table dimCountry  
(  
 country\_id int not null auto\_increment,  
 country varchar(40),  
 primary key (country\_id)  
);  
  
create table dimCity  
(  
 city\_id int not null auto\_increment,  
 country\_id int not null,  
 city varchar(40),  
 primary key (city\_id),  
 foreign key (country\_id) references dimCountry (country\_id)  
);  
  
create table dimStadium  
(  
 stadium\_id int not null auto\_increment,  
 source\_id int default null,  
 city\_id int not null,  
 name varchar(50),  
 capacity varchar(50),  
 start\_date date default null,  
 end\_date date default null,  
 primary key (stadium\_id),  
 foreign key (city\_id) references dimCity (city\_id)  
);  
  
create table dimManager  
(  
 manager\_id int not null auto\_increment,  
 first\_name varchar(50),  
 last\_name varchar(50),  
 nationality varchar(40),  
 date\_of\_birth date,  
 team varchar(40),  
 primary key (manager\_id)  
);  
  
create table dimSeason  
(  
 season\_id int not null auto\_increment,  
 season varchar(9),  
 primary key (season\_id)  
);  
  
create table dimDate  
(  
 date\_id int not null auto\_increment,  
 year int,  
 month int,  
 day int,  
 primary key (date\_id)  
);  
  
create table dimTeams  
(  
 team\_id int not null,  
 team\_name varchar(50),  
 country varchar(40),  
 home\_stadium varchar(40),  
 primary key (team\_id)  
);  
  
create table factMatches  
(  
 match\_id int not null 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,  
 isPenalty tinyint(1),  
 attendance int,  
 primary key (match\_id),  
 foreign key (date\_id) references dimDate (date\_id),  
 foreign key (home\_team\_id) references dimTeams (team\_id),  
 foreign key (away\_team\_id) references dimTeams (team\_id),  
 foreign key (season\_id) references dimSeason (season\_id),  
 foreign key (home\_manager\_id) references dimManager (manager\_id),  
 foreign key (away\_manager\_id) references dimManager (manager\_id),  
 foreign key (stadium\_id) references dimStadium (stadium\_id)  
);

Скрипти для заповнення основного сховища та зв’язування даних між собою:

use stage;  
  
  
insert into main\_warehouse.dimteams(team\_id, team\_name, country, home\_stadium)  
select distinct id, team\_name, country, home\_stadium  
from teams;  
  
insert into main\_warehouse.dimseason(season)  
select distinct season  
from matches;  
  
insert into main\_warehouse.dimdate(year, month, day)  
select distinct *year*(date\_time), *month*(date\_time), *day*(date\_time)  
from matches;  
  
insert into main\_warehouse.dimmanager(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\_warehouse.dimcountry(country)  
select distinct country  
from stadiums;  
  
insert into main\_warehouse.dimcity(country\_id, city)  
select distinct dc.country\_id, city  
from stadiums  
 join main\_warehouse.dimcountry dc on dc.country = stadiums.country;  
  
insert into main\_warehouse.dimstadium(city\_id, name, capacity)  
select distinct city\_id, name, capacity  
from stadiums  
 join main\_warehouse.dimcity ds on ds.city = stadiums.city;  
  
insert into main\_warehouse.factmatches(date\_id, home\_team\_id, away\_team\_id, season\_id, home\_manager\_id, away\_manager\_id,  
 stadium\_id,  
 home\_team\_score, away\_team\_score, isPenalty, attendance)  
select d.date\_id,  
 t.team\_id,  
 t2.team\_id,  
 s.season\_id,  
 m1.manager\_id,  
 m2.manager\_id,  
 st.stadium\_id,  
 home\_team\_score,  
 away\_team\_score,  
 penalty\_shoot\_out,  
 attendance  
from matches m  
 join main\_warehouse.dimdate d  
 on *year*(m.date\_time) = d.year and *month*(m.date\_time) = d.month and *day*(m.date\_time) = d.day  
 join main\_warehouse.dimteams t on m.home\_team = t.team\_name  
 join main\_warehouse.dimteams t2 on m.away\_team = t2.team\_name  
 join main\_warehouse.dimseason s on m.season = s.season  
 join main\_warehouse.dimstadium st on m.stadium = st.name  
 join main\_warehouse.dimmanager m1 on m.home\_team = m1.team  
 join main\_warehouse.dimmanager m2 on m.away\_team = m2.team;

Скрипт для реалізації можливості slowly changing dimension:

use main\_warehouse;  
  
drop procedure if exists *slow\_change\_stadiums*;  
  
delimiter //  
create procedure *slow\_change\_stadiums*(old\_name varchar(50), new\_name varchar(50))  
begin  
 declare old\_id int default null;  
 declare old\_city\_id int;  
 declare old\_capacity varchar(50);  
  
 select stadium\_id, city\_id, capacity  
 into old\_id, old\_city\_id, old\_capacity  
 from dimstadium  
 where name = old\_name;  
 if old\_id is null then  
 signal sqlstate '45000' set message\_text = 'The old name of the stadium does not exist';  
 else  
 insert into dimstadium (source\_id, city\_id, name, capacity, start\_date)  
 value (old\_id, old\_city\_id, new\_name, old\_capacity, *current\_date*);  
  
 update dimstadium  
 set end\_date = *CURRENT\_DATE* where stadium\_id = old\_id;  
  
 update dimteams  
 set home\_stadium = new\_name  
 where home\_stadium = old\_name;  
 end if;  
end //  
  
delimiter ;  
  
call *slow\_change\_stadiums*('Olimpiyskyi', 'Olimpiyskyi Ukraine');  
call *slow\_change\_stadiums*('Olimpiyskyi Ukraine', 'Olimpiyskyi Kyiv');

Скрипт для можливості завантаження нових даних до існуючих:

use main\_warehouse;  
  
insert into dimseason (season)  
select distinct season  
from stage.matches m  
where not *exists*(select season from dimseason ds where ds.season = m.season);  
  
insert into dimdate (year, month, day)  
select distinct *year*(m.date\_time), *month*(m.date\_time), *day*(m.date\_time)  
from stage.matches m  
where not *exists*(select *year*(m.date\_time), *month*(m.date\_time), *day*(m.date\_time)  
 from dimdate dd  
 where dd.year = *year*(m.date\_time)  
 and dd.month = *month*(m.date\_time)  
 and dd.day = *day*(m.date\_time));  
  
insert into dimmanager (first\_name, last\_name, nationality, date\_of\_birth, team)  
select distinct first\_name, last\_name, nationality, date\_of\_birth, team  
from stage.managers mn  
where not *exists*(select dm.first\_name, dm.last\_name, dm.team  
 from dimmanager dm  
 where dm.first\_name = mn.first\_name  
 and dm.last\_name = mn.last\_name  
 and dm.team = mn.team);  
  
insert into dimteams (team\_name, country, home\_stadium)  
select distinct team\_name, country, home\_stadium  
from stage.teams t  
where not *exists*(select dt.team\_name from dimteams dt where t.team\_name = dt.team\_name);  
  
insert into dimcountry (country)  
select distinct country  
from stage.stadiums st  
where not *exists*(select dc.country from dimcountry dc where dc.country = st.country);  
  
insert into dimcity (country\_id, city)  
select distinct country\_id, city  
from stage.stadiums st  
 join dimcountry dc on st.country = dc.country  
where not *exists*(select dc.city from dimcity dc where dc.city = st.city);  
  
insert into dimstadium (source\_id, city\_id, name, capacity, start\_date, end\_date)  
select distinct null, dc.city\_id, st.name, st.capacity, null, null  
from stage.stadiums st  
 join dimcity dc on st.city = dc.city  
where not *exists*(select ds.name from dimstadium ds where st.name = ds.name);  
  
  
insert into factmatches (date\_id, home\_team\_id, away\_team\_id, season\_id, home\_manager\_id, away\_manager\_id, stadium\_id,  
 home\_team\_score, away\_team\_score, isPenalty, attendance)  
select distinct d.date\_id,  
 t.team\_id,  
 t2.team\_id,  
 s.season\_id,  
 m1.manager\_id,  
 m2.manager\_id,  
 st.stadium\_id,  
 home\_team\_score,  
 away\_team\_score,  
 penalty\_shoot\_out,  
 attendance  
from stage.matches m  
 join main\_warehouse.dimdate d  
 on *year*(m.date\_time) = d.year and *month*(m.date\_time) = d.month and *day*(m.date\_time) = d.day  
 join main\_warehouse.dimteams t on m.home\_team = t.team\_name  
 join main\_warehouse.dimteams t2 on m.away\_team = t2.team\_name  
 join main\_warehouse.dimseason s on m.season = s.season  
 join main\_warehouse.dimstadium st on m.stadium = st.name  
 join main\_warehouse.dimmanager m1 on m.home\_team = m1.team  
 join main\_warehouse.dimmanager m2 on m.away\_team = m2.team  
where not *exists*(select fm.date\_id, fm.home\_team\_id, fm.away\_team\_id  
 from factmatches fm  
 where fm.date\_id = d.date\_id  
 and fm.home\_team\_id = t.team\_id  
 and fm.away\_team\_id = t2.team\_id);

Висновок

У цьому комп’ютерному практикуму ми ознайомились з можливістю проектування сховища даних, проходячи етапи створення stage зони для завантаження даних та створення основного сховища для розподілення даних за певною логікою зі зв’язками між ними. Було реалізовано можливості slowly changing dimension та завантаження нових даних до вже існуючих. Попередньо дані були оброблені, щоб привести їх до потрібного формату.