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 2023

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:

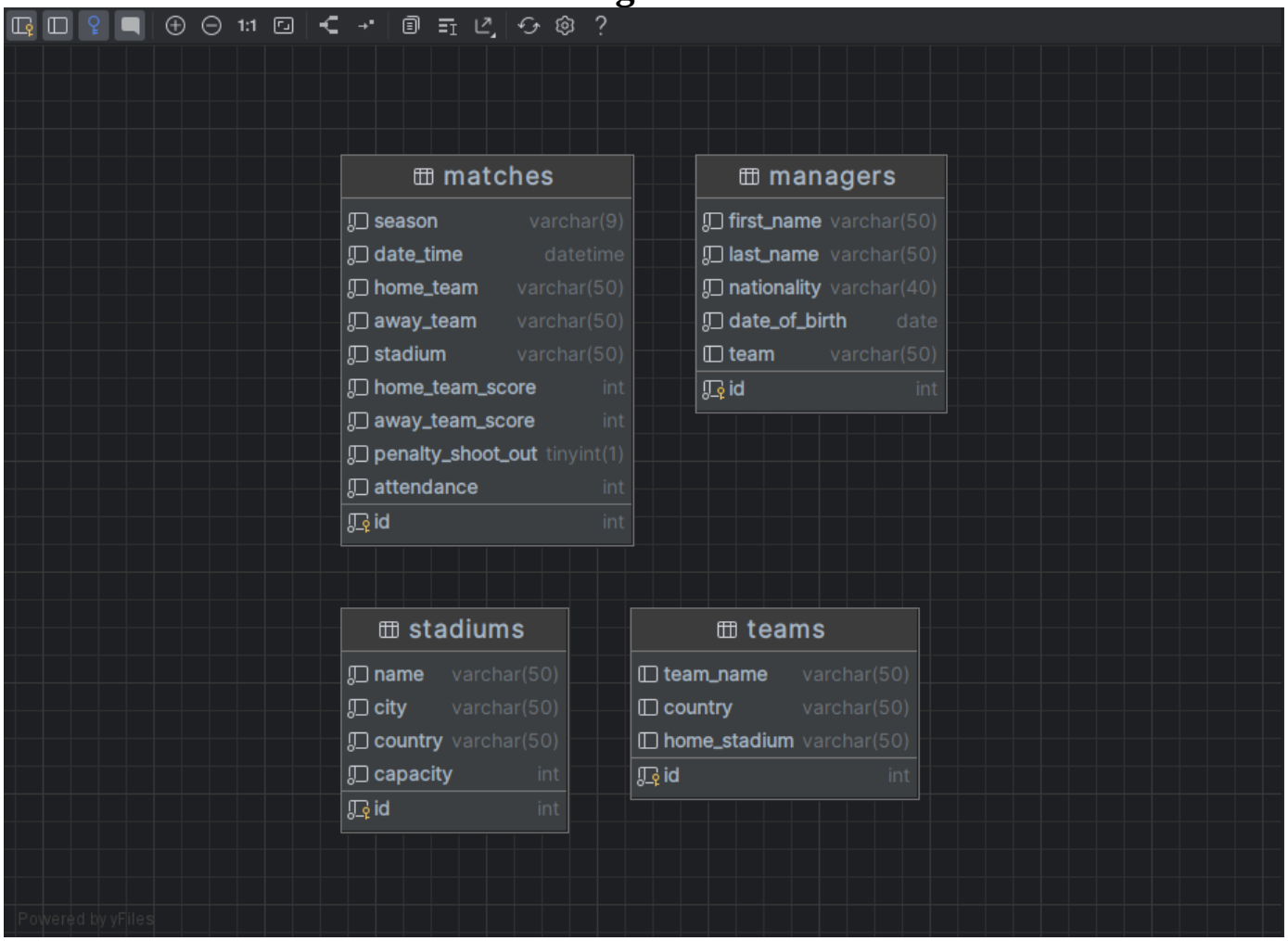
main.py

```
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



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:

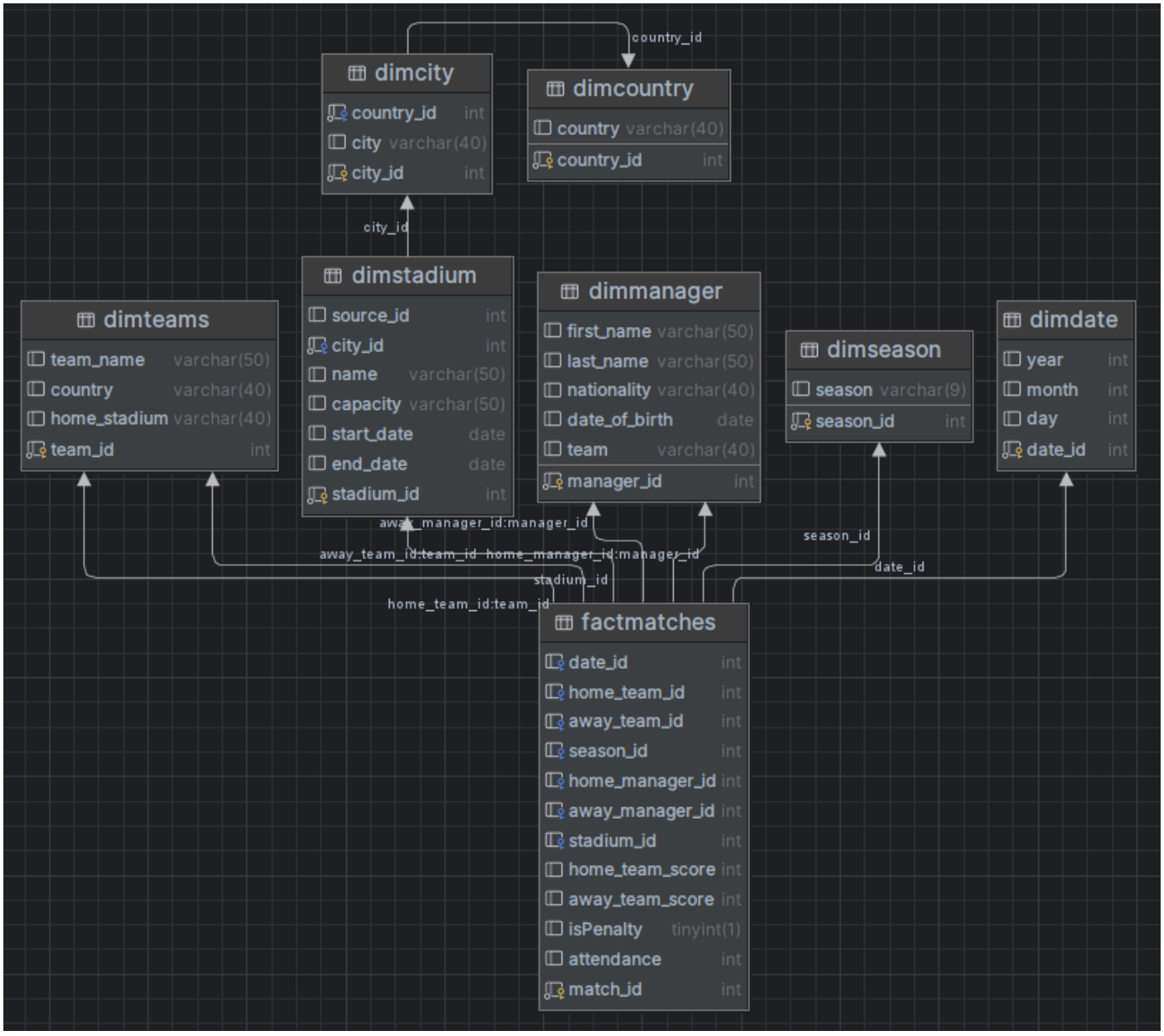
```
drop
    database if exists stage;
create
    database stage;

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



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.