Dokumentacja projektu wykonywanego w ramach zajęć BAZY DANYCH I

System przechowywania informacji o rozgrywkach piłki nożnej

I. Projekt koncepcji, założenia

1. Zdefiniowanie tematu projektu:

   Stworzyłem aplikację mającą na celu przechowywanie informacji o rozgrywkach piłki nożnej. Przechowuję informację o lidze, klubach, piłkarzach, meczach oraz wydarzeniach podczas meczy(gol, kartka, zmiana). Dzięki aplikacji mamy podgląd do tabeli ligowych, listy piłkarzy, listy meczy i wydarzeń podczas nich.

2. Analiza wymagań użytkownika:

   Baza danych umożliwia wprowadzanie do bazy danych drużynę, piłkarza, mecz i wydarzenia podczas meczu. Przy wprowadzaniu jest sprawdzana poprawność danych. Od razu po wprowadzeniu nowych danych automatycznie aktualizowana jest reszta tabel.

3. Zaprojektowanie funkcji:

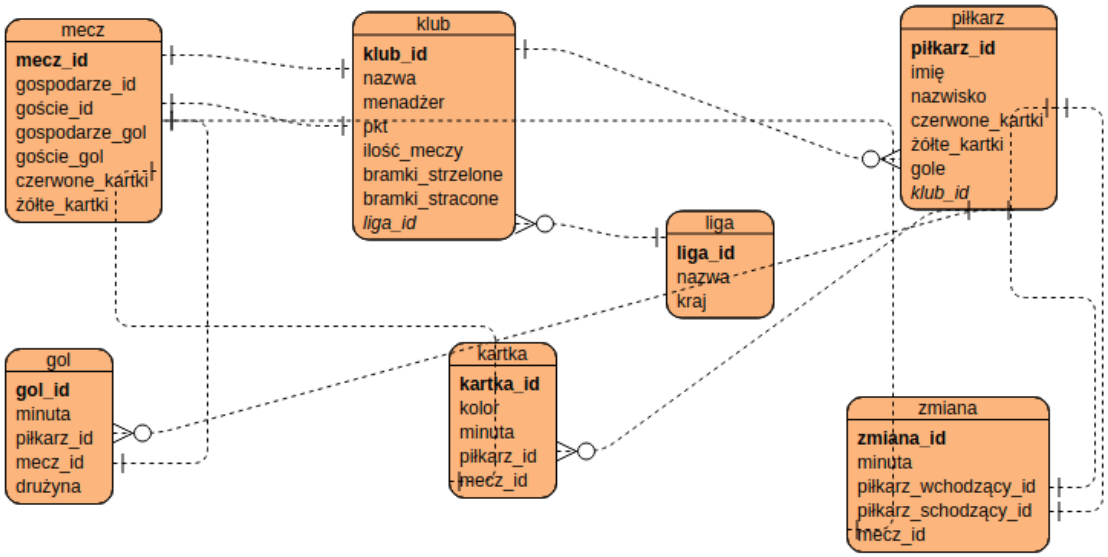
   - dodajZmiane – dodaje zmianę
   - dodajGola – dodaje gola
   - dodajPiłkarza – dodaje piłkarza
   - dodajKartke – dodaje kartkę
   - dodajKlub – dodaje klub
   - wyswietlTabele – wyswietla w formie tabeli informacje
   - wyswietlTabeleZmiana - wyswietla w formie tabeli informacje
   - wyswietlTabeleKartka - wyswietla w formie tabeli informacje
   - wyswietlTabeleGola - wyswietla w formie tabeli informacje
   - fillComboBox (fillComboBox_1 itd.) - wypełnia comboBoxy

II. Projekt diagramów (konceptualny)

4. Budowa i analiza diagramu przepływu danych (DFD):
5. Zdefiniowanie encji (obiektów) oraz ich atrybutów:

   - liga (liga_id SERIAL PRIMARY KEY, nazwa VARCHAR(50), kraj VARCHAR(50));
   - klub (klub_id SERIAL PRIMARY KEY, nazwa VARCHAR(50), menadżer VARCHAR(50), pkt INTEGER, ilość_meczy INTEGER, bramki_strzelone INTEGER, bramki_stracone INTEGER, liga_id INTEGER);
   - piłkarz(piłkarz_id SERIAL PRIMARY KEY, imię VARCHAR(50), nazwisko VARCHAR(50), czerwone_kartki INTEGER, żółte_kartki INTEGER, klub_id INTEGER, gole INTEGER);
   - mecz (mecz_id SERIAL PRIMARY KEY, gospodarze_id INTEGER, goście_id INTEGER, gospodarze_gol INTEGER, goście_gol INTEGER, czerwone_kartki INTEGER, żółte_kartki INTEGER);
   - kartka (kartka_id SERIAL primary key, kolor VARCHAR(10), minuta INTEGER, piłkarz_id INTEGER, mecz_id INTEGER);
   - gol(gol_id SERIAL primary key, minuta INTEGER, drużyna VARCHAR(50), piłkarz_id INTEGER, mecz_id INTEGER);
   - zmiana (zmiana_id SERIAL PRIMARY KEY, minuta INTEGER, piłkarz_wchodzący_id INTEGER, piłkarz_schodzący_id INTEGER, mecz_id INTEGER);

6. Zaprojektowanie relacji pomiędzy encjami:



III. Projekt logiczny

7. Projektowanie tabel, kluczy, indeksów:

Tworzenie tabel (klucze główne) :

```
CREATE TABLE klub ( klub_id SERIAL PRIMARY KEY, nazwa VARCHAR(50),
    menadżer VARCHAR(50), pkt INTEGER, ilość_meczy INTEGER,
    bramki_strzelone INTEGER, bramki_stracone INTEGER, liga_id INTEGER );
CREATE TABLE piłkarz( piłkarz_id SERIAL PRIMARY KEY, imię VARCHAR(50),
    nazwisko VARCHAR(50), czerwone_kartki INTEGER, żółte_kartki INTEGER,
    klub_id INTEGER, gole INTEGER );
CREATE TABLE mecz ( mecz_id SERIAL PRIMARY KEY, gospodarze_id INTEGER,
    goście_id INTEGER, gospodarze_gol INTEGER, goście_gol INTEGER,
    czerwone_kartki INTEGER, żółte_kartki INTEGER );
CREATE TABLE kartka ( kartka_id SERIAL PRIMARY KEY, kolor VARCHAR(10),
    minuta INTEGER, piłkarz_id INTEGER, mecz_id INTEGER );
CREATE TABLE gol( gol_id SERIAL PRIMARY KEY, minuta INTEGER, drużyna
        VARCHAR(50), piłkarz_id INTEGER, mecz_id INTEGER );
CREATE TABLE zmiana ( zmiana_id SERIAL PRIMARY KEY, minuta INTEGER,
    piłkarz_wchodzący_id INTEGER, piłkarz_schodzący_id INTEGER, mecz_id
    INTEGER);
CREATE TABLE liga ( liga_id SERIAL PRIMARY KEY, nazwa VARCHAR(50), kraj
    VARCHAR(50) );
```

Klucze obce :

```
ALTER TABLE gol ADD FOREIGN KEY(piłkarz_id) REFERENCES piłkarz(piłkarz_id);
ALTER TABLE gol ADD FOREIGN KEY(mecz_id) REFERENCES mecz(mecz_id);
ALTER TABLE kartka ADD FOREIGN KEY(piłkarz_id) REFERENCES
    piłkarz(piłkarz_id);
ALTER TABLE kartka ADD FOREIGN KEY(mecz_id) REFERENCES mecz(mecz_id);
```

```
ALTER TABLE zmiana ADD FOREIGN KEY(mecz_id) REFERENCES mecz(mecz_id);
ALTER TABLE zmiana ADD FOREIGN KEY(piłkarz_wchodzący_id) REFERENCES
    piłkarz(piłkarz_id);
ALTER TABLE zmiana ADD FOREIGN KEY(piłkarz_schodzący_id) REFERENCES
    piłkarz(piłkarz_id);
ALTER TABLE klub ADD FOREIGN KEY(liga_id) REFERENCES liga(liga_id);
ALTER TABLE mecz ADD FOREIGN KEY(gospodarze_id) REFERENCES
    klub(klub_id);
ALTER TABLE mecz ADD FOREIGN KEY(goście_id) REFERENCES klub(klub_id);
```

Wypełnianie tabel jest bardzo długie więc pozwolę sobie go tutaj nie umieścić. Wszystkie tabele zawierają klucz główny który je jednoznacznie opisuje.

Tabela liga – reprezentuje ligę.

Tabela klub – reprezentuje pojedynczy klub, ma relacje 1 do N z tabelą liga, ponieważ w jednej lidze może a nawet powinno być wiele klubów.

Tabela piłkarz – reprezentuje piłkarza, który posiada relacje 1 do N z tabelą klub, ponieważ w klubie może być wielu piłkarzy.

Tabela mecz – reprezentuje mecz, posiada 2 relacje 1 do 1 z klub, ponieważ musi mieć odwołanie do drużyny gospodarzy i gości.

Tabela gol – reprezentuje gola, posiada po jednej relacji 1 do 1 z piłkarz oraz mecz, ponieważ musi mieć odwołanie do piłkarza który bramkę strzelił i do meczu w którym padła.

Tabela zmiana – reprezentuje zmianę, posiada 2 relacje 1 do 1 z piłkarz i 1 relacje 1 do 1 z mecz, ponieważ musi wiedzieć jaki piłkarz wszedł, a jaki zszedł z boiska oraz w jakim było to meczu;

Tabela kartka – reprezentuje kartkę, posiada po jednej relacji 1 do 1 z piłkarz oraz mecz podobnie jak gol.

8. Słowniki danych:
9. Analiza zależności funkcyjnych i normalizacja tabel (dekompozycja do 3NF ewentualnie BCNF):
10. Denormalizacja struktury tabel:
11. Zaprojektowanie operacji na danych:

**** funkcje dla triggera, która przy dodaniu lub update „meczu" update inny tabelki odpowiednimi wartościami ****

```
CREATE OR REPLACE FUNCTION mecz()
RETURNS TRIGGERAS $$
DECLARE
 quantity INTEGER;
 counter INTEGER;
 mecz_counter INTEGER;
 temp INTEGER;
 tmp_pkt INTEGER;
BEGIN
 quantity := (SELECT MAX(klub_id) FROM klub);
 counter := 1;
 LOOP
```

```
        tmp_pkt := 0;
        EXIT WHEN counter > quantity;
        mecz_counter := ( ( SELECT COUNT(*) FROM mecz WHERE gospodarze_id =
counter ) + ( SELECT COUNT(*) FROM mecz WHERE goście_id = counter ) );
        UPDATE klub SET ilość_meczy = mecz_counter FROM mecz WHERE
klub.klub_id = counter;

        mecz_counter := ( (SELECT COALESCE(SUM(gospodarze_gol),0) FROM mecz
WHERE gospodarze_id = counter) + (SELECT COALESCE(SUM(goście_gol),0) FROM
mecz WHERE goście_id = counter) );
        UPDATE klub SET bramki_strzelone = mecz_counter FROM mecz WHERE
klub.klub_id = counter;
        mecz_counter := ( ( SELECT COALESCE(SUM(goście_gol),0) FROM mecz
WHERE gospodarze_id = counter ) + ( SELECT COALESCE(SUM(gospodarze_gol),0)
FROM mecz WHERE goście_id = counter ) );
        UPDATE klub SET bramki_stracone = mecz_counter FROM mecz WHERE
klub.klub_id = counter;

        mecz_counter := ( (SELECT COUNT(*) FROM mecz WHERE ( gospodarze_id =
counter AND (gospodarze_gol - goście_gol > 0) ) ) + (SELECT COUNT(*) FROM mecz
WHERE (goście_id = counter AND (gospodarze_gol - goście_gol < 0))) );
        tmp_pkt := tmp_pkt + mecz_counter * 3;
        mecz_counter := ( ( SELECT COUNT(*) FROM mecz WHERE ( gospodarze_id =
counter AND (gospodarze_gol - goście_gol = 0 ) ) ) + ( SELECT COUNT(*) FROM mecz
WHERE ( goście_id = counter AND (gospodarze_gol - goście_gol = 0 ) ) ) );
        tmp_pkt := tmp_pkt + mecz_counter;
        UPDATE klub SET pkt = tmp_pkt FROM mecz WHERE klub.klub_id = counter;

        counter := counter + 1;
 END LOOP;

 RETURN NULL;
END;
$$
LANGUAGE plpgsql;

CREATE TRIGGER mecz_trigger AFTER INSERT OR UPDATE ON mecz EXECUTE
PROCEDURE mecz();
```

**** funkcje dla triggera, która przy dodaniu lub update „gol" update inny tabelki odpowiednimi wartościami ****

```
CREATE OR REPLACE FUNCTION gol()
RETURNS TRIGGERAS $$
DECLARE
 quantity INTEGER;
 counter INTEGER;
 gol_counter INTEGER;
```

```
BEGIN
 quantity := (SELECT MAX(mecz_id) FROM mecz);
 counter := 1;
 LOOP
        EXIT WHEN counter > quantity;
        gol_counter := ( SELECT COUNT(*) FROM gol WHERE ( (mecz_id = counter)
AND (drużyna = 'gospodarz') ));
        UPDATE mecz SET gospodarze_gol = gol_counter FROM gol WHERE
mecz.mecz_id = counter;
        gol_counter := ( SELECT COUNT(*) FROM gol WHERE ( (mecz_id = counter)
AND (drużyna = 'gość') ));
        UPDATE mecz SET goście_gol = gol_counter FROM gol WHERE mecz.mecz_id =
counter;
        counter := counter + 1;
 END LOOP;


 quantity := (SELECT MAX(piłkarz_id) FROM piłkarz);
 counter := 1;
 LOOP
        EXIT WHEN counter > quantity;
        gol_counter := ( SELECT COUNT(*) FROM gol WHERE piłkarz_id = counter );
        UPDATE piłkarz SET gole = gol_counter FROM gol WHERE piłkarz.piłkarz_id =
counter;
        counter := counter + 1;
 END LOOP;
 RETURN NULL;
END;
$$
LANGUAGE plpgsql;

CREATE TRIGGER gol_trigger AFTER INSERT OR UPDATE ON gol EXECUTE
PROCEDURE gol();
```

**** funkcje dla triggera, która przy dodaniu lub update „kartka” update inny tabelki odpowiednimi wartościami ****

```
CREATE OR REPLACE FUNCTION kartka()
RETURNS TRIGGERAS $$
DECLARE
 quantity INTEGER;
 counter INTEGER;
 kartka_counter INTEGER;
BEGIN
 quantity := (SELECT MAX(mecz_id) FROM mecz);
 counter := 1;
 LOOP
        EXIT WHEN counter > quantity;
```

```
        kartka_counter := ( SELECT COUNT(*) FROM kartka WHERE ( (mecz_id =
counter) AND (kolor = 'czerwona') ));
        UPDATE mecz SET czerwone_kartki = kartka_counter FROM kartka WHERE
mecz.mecz_id = counter;
        kartka_counter := ( SELECT COUNT(*) FROM kartka WHERE ( (mecz_id =
counter) AND (kolor = 'żółta') ));
        UPDATE mecz SET żółte_kartki = kartka_counter FROM kartka WHERE
mecz.mecz_id = counter;
        counter := counter + 1;
 END LOOP;

 quantity := (SELECT MAX(piłkarz_id) FROM piłkarz);
 counter := 1;
 LOOP
        EXIT WHEN counter > quantity;
        kartka_counter := ( SELECT COUNT(*) FROM kartka WHERE ( (piłkarz_id =
counter) AND (kolor = 'czerwona') ));
        UPDATE piłkarz SET czerwone_kartki = kartka_counter FROM kartka WHERE
piłkarz.piłkarz_id = counter;
        kartka_counter := ( SELECT COUNT(*) FROM kartka WHERE ( (piłkarz_id =
counter) AND (kolor = 'żółta') ));
        UPDATE piłkarz SET żółte_kartki = kartka_counter FROM kartka WHERE
piłkarz.piłkarz_id = counter;
        counter := counter + 1;
 END LOOP;
 RETURN NULL;
END;
$$
LANGUAGE plpgsql;

CREATE TRIGGER kartka_trigger AFTER INSERT OR UPDATE ON kartka EXECUTE
PROCEDURE kartka();
```

**** widoki stworzone, żeby była dobrze przedstawiona tabela ligowa dla danej ligi****

```
CREATE VIEW premier_league AS SELECT nazwa, menadżer, pkt, ilość_meczy,
bramki_strzelone, bramki_stracone FROM klub WHERE liga_id = 1 ORDER BY pkt
DESC;
CREATE VIEW la_liga AS SELECT nazwa, menadżer, pkt, ilość_meczy, bramki_strzelone,
bramki_stracone FROM klub WHERE liga_id = 2 ORDER BY pkt DESC;
CREATE VIEW ligue_1 AS SELECT nazwa, menadżer, pkt, ilość_meczy, bramki_strzelone,
bramki_stracone FROM klub WHERE liga_id = 3 ORDER BY pkt DESC;
CREATE VIEW serie_A AS SELECT nazwa, menadżer, pkt, ilość_meczy,
bramki_strzelone, bramki_stracone FROM klub WHERE liga_id = 4 ORDER BY pkt
DESC;
CREATE VIEW bundesliga AS SELECT nazwa, menadżer, pkt, ilość_meczy,
bramki_strzelone, bramki_stracone FROM klub WHERE liga_id = 5 ORDER BY pkt
DESC;
```

CREATE VIEW ekstraklasa AS SELECT nazwa, menadżer, pkt, ilość_meczy, bramki_strzelone, bramki_stracone FROM klub WHERE liga_id = 6 ORDER BY pkt DESC;

IV. Projekt funkcjonalny

12. Interfejsy do prezentacji, edycji i obsługi danych:

- formularz „dodaj klub” - pobiera nazwę klubu, nazwę menadżera i liga_id z wybranej z listy ligi.
- formularz „dodaj piłkarz” - pobiera imię, nazwisko piłkarza i klub_id z wybranego z listy klubu.
- formularz „dodaj mecz” - pobiera gospodarz_id oraz gość_id z wybranych klubów z list klubów. Zaznaczę, że po wyborze gospodarza możemy wybrać gościa tylko z tej samej ligi.
- formularz „dodaj gola” - pobiera mecz_id, informację czy strzelił bramkę gospodarz czy gość oraz piłkarz_id z list. Pobiera również minutę wprowadzoną ręcznie. Tutaj też po wybraniu meczy i gospodarz\gość możemy wybrać strzelca tylko z danego klubu.
- formularz „dodaj kartkę” - podobnie jak „dodaj gola” tylko kartkę
- formularz „dodaj zmianę” - podobnie jak „dodaj gola” oraz „dodaj kartkę” tylko zamiast jednego piłkarz wybieramy 2, schodzącego i wchodzącego na boisko.

13. Wizualizacja danych:

W formie tabelki wypisywane są: tabele ligowe, kluby, piłkarze, mecze, gole, zmiany, kartki.
W listach są mecze, kluby, piłkarze wykorzystywane przez formularze.

14. Zdefiniowanie panelu sterowania aplikacji:

Poruszamy się po menu w lewym górnym rogu. Wszystkie operacje są opisane.

15. Makropolecenia:

V. Dokumentacja

16. Wprowadzanie danych:

Większość jest pobierana z list, tylko nazwy własne i wartości własne są wprowadzane ręcznie
Automatycznie: update wszystkich tabel po dodaniu danego wydarzenia.

17. Dokumentacja użytkownika:

Program jest bardzo łatwy w obsłudze, wszystko odbywa się przy pomocy rozwijanego menu w lewym górnym rogu, każdy przycisk jest opisany. Możemy za pomocą menu zobaczyć daną tabelę ligową, listy klubów, meczy itd. Możemy się dostać do formularzy dodawania, które po dodaniu przenoszą od raz do listy tego typu który właśnie dodaliśmy.

18. Opracowanie dokumentacji technicznej:

Wygenerowane zostały pliki za pomocą javadocs, są umieszczone w folderze ‘docs’;

19. Wykaz literatury:

https://tableplus.com/