Politechnika Warszawska
Wydział Elektroniki i Technik Informacyjnych

Wprowadzenie do Baz Danych - Projekt

System biurowych zakładów wzajemnych

Zdający:
Robert Wojtaś
Jakub Sikora

Prowadzący:
dr inż. Marcin Kowalczyk

Warszawa, 3 stycznia 2019

Spis treści

Spis tabel

Spis rysunków

1. Zakres i cel projektu

1.1. Cel projektu

Celem projektu jest poprawne zaprojektowanie relacyjnej bazy danych oraz jej fizyczna implementacja przy użyciu systemu Oracle. W trakcie projektowania należy odpowiednio podzielić fazy projektowania na poziom konceptualny oraz logiczny a także doprowadzić projekt bazy do trzeciej postaci normalnej.

1.2. Założenia projektowe

Realizowany projekt dotyczy biurowego systemu obstawiania meczów na dużych turniejach sportowych. System ten zajmuje się zbieraniem zakładów od swoich użytkowników oraz prowadzeniem statystyk. Oferuje swoim użytkownikom możliwość zakładania się na wyniki spotkań rozgrywanych w ramach uprzednio zdefiniowanych turniejów oraz podliczaniem wyników wedle ustalonego algorytmu punktowania.

W tym celu, system prowadzi bazę danych która zbiera informacje o spotkaniach oraz dostępnych turniejach. Każdy uczestnik gry próbuje przewidzieć dokładny wynik spotkania i w zależności od poprawności, otrzymuje 3, 1 albo 0 punktów. Trzy punkty gracz otrzymuje, gdy padnie dokładnie obstawiony przez niego wynik. Gracz otrzymuje jeden punkt, gdy końcowy rezultat jest taki sam jak obstawiony z dokładnością do zdobytych przez drużynę punktów. Przykładowo, gracz obstawił że meczu piłki nożnej zakończy się rezultatem 3:1 a mecz zakończył się wynikiem 4:2. Gdy gracz nie trafi w rezultat, nie otrzymuje punktów.

W celu ułatwienia graczom podejmowanie decyzji, system oferuje szeroką gamę statystyk prowadzonych w ramach turniejów. Baza przechowuje informacje o nadchodzących spotkaniach pomiędzy dwoma drużynami, o zawodnikach występujących w tych drużynach, o sędziach przewidzianych do prowadzania danego spotkania a także o planowanym miejscu rozegrania spotkania.

Każdy użytkownik może grać niezależnie w kilku różnych turniejach i dodawać zakłady na dowolną ilość spotkań z takim zastrzeżeniem, że nie wolno dodać zakładu na rozpoczęte już spotkania. Dodatkowo, każdy turniej oferuje specjalne zakłady długoterminowe dotyczące ostatecznego zwycięzcy i najbardziej wartościowego zawodnika turnieju. Zakłady te są warte odpowiednio więcej punktów. Po zakończeniu wszystkich spotkań z danego turnieju, wybierany jest zwycięzca na podstawie zdobytej liczby punktów.

2. Definicja systemu

2.1. Perspektywy użytkowników

W ramach systemu zdefiniowaliśmy trzy typy potencjalnych użytkowników:

1. Użytkownik - uczestnik gry, dodaje zakłady nierozpoczęte jeszcze spotkania w ramach turniejów na które się wcześniej zapisał

2. Statystyk - moderator gry, dodaje informacje o drużynach, sędziach, stadionach i trenerach oraz na bieżąco uzupełnia wyniki spotkań i turniejów

3. Administrator - główny moderator systemu, tworzy konta użytkownikom oraz przywraca dostępy

2.2. Zdefiniowane funkcjonalności

Do zdefiniowania funkcjonalności posłużyliśmy się metodyką User Stories znanych z metodyki Agile. Wcieliśmy się w rolę każdego z użytkowników i opisaliśmy potrzebne funkcjonalności według znanego schematu:

Jako osoba ...,
potrzebuję/chcę takiej funkcjonalności ...,
ponieważ pozwoli mi to ...

Na podstawie tak opisanych funkcjonalności, w łatwy sposób mogliśmy zdefiniować potrzebne transakcje w systemie. Dodatkowo, taki sposób opisu funkcjonalności już na etapie projektowania sprawdza ich przydatność.

2.2.1. Użytkownik systemu

Zapis na turniej

Jako użytkownik, potrzebuję mieć możliwość zapisania się na turniej, ponieważ wtedy będę mógł wziąć udział w grze.

Dodawanie zakładu na spotkanie

Jako użytkownik, potrzebuję móc dodawać nowe zakłady na spotkania które jeszcze się nie rozpoczęły, ponieważ pozwoli mi to na zdobywanie punktów.

Edycja zakładu na spotkanie

Jako użytkownik, potrzebuję móc edytować swoje zakłady na spotkania które jeszcze się nie rozpoczęły, ponieważ pozwoli mi to na zmianę zdania i poprawienie swojego zakładu.

Dodawanie zakładu długoterminowego

Jako użytkownik, potrzebuję móc dodawać zakład długoterminowy na zwycięzce turnieju na który się zapisałem a który jeszcze się nie rozpoczął, ponieważ dzięki temu będę mógł zdobyć więcej punktów.

Edycja zakładu długoterminowego

Jako użytkownik, potrzebuję móc edytować zakład długoterminowego na zwycięzce turnieju na który się zapisałem a który jeszcze się nie rozpoczął, ponieważ pozwoli mi to na zmianę zdania i poprawienie swojego zakładu.

Podgląd tabeli

Jako użytkownik, potrzebuję móc sprawdzać który jestem w tabeli wyników, ponieważ potrzebuję informacji o ewentualnym zwycięstwie.

2.2.2. Statystyk

Dodawanie turniejów

Jako statystyk, potrzebuję możliwości dodawania turniejów, ponieważ wtedy użytkownicy będą mogli się na nie zapisywać.

Dodawanie meczów

Jako statystyk, potrzebuję możliwości dodawania meczów w ramach turniejów, ponieważ wtedy użytkownicy będą mogli się próbować obstawić ich wynik.

Dodawanie drużyn

Jako statystyk, potrzebuję możliwości dodawania drużyn do systemu, ponieważ wtedy będę mógł je podpinać pod turnieje.

Podpinanie drużyn do turniejów

Jako statystyk, potrzebuję możliwości dodawania drużyn jako uczestników turniejów, ponieważ wtedy będę mógł dodawać mecze do turniejów pomiędzy nimi.

Dodawanie stadionów

Jako statystyk, potrzebuję możliwości dodawania stadionów do systemu, ponieważ wtedy będę mógł podpinać stadiony do meczów.

Podpinanie stadionów

Jako statystyk, potrzebuję możliwości podpinania stadionów do meczów, ponieważ wtedy użytkownicy będą więcej wiedzieli o meczu co pozwoli im lepiej obstawić.

Dodawanie sędziów

Jako statystyk, potrzebuję możliwości dodawania sędziów, ponieważ wtedy będę mógł podpinać sędziów głównych do spotkań.

Podpinanie sędziów

Jako statystyk, potrzebuję możliwości podpinania sędziów do meczów, ponieważ wtedy użytkownicy będą więcej wiedzieli o meczu co pozwoli im lepiej obstawić.

Dodawanie wyników meczów

Jako statystyk, potrzebuję możliwości dodawania wyników zakończonych już spotkań, ponieważ wtedy użytkownicy będą mogli zweryfikować swoje zakłady.

Dodawania zwycięzców turniejów

Jako statystyk, potrzebuję możliwości dodawania zwycięzców turniejów, ponieważ wtedy użytkownicy będą mogli zweryfikować swoje zakłady długoterminowe.

Aktualizowanie składów drużyn

Jako statystyk, potrzebuję możliwości aktualizowania składów drużyn, ponieważ wtedy statystyki będą aktualne.

2.2.3. Administrator

Dodawanie użytkowników

Jako administrator, potrzebuję możliwości tworzenia kont dla użytkowników, ponieważ wtedy będą mogli wziąc udział w grze.

Zmiana hasła użytkownika

Jako administrator, potrzebuję możliwości zmiany hasła użytkownika, ponieważ wtedy będę mógł przywrócić dostęp tym którzy zapomnieli hasła.

3. Model konceptualny

Modelem konceptualnym nazywamy pewną reprezentację obiektów świata rzeczywistego w uniwersalnym modelu niezależnym od implementacji [1]. Poprawny model konceptualny jest dobrym punktem wyjścia, ponieważ pozwala na przemyślenie jakie dane tak naprawdę chcemy przechowywać w projektowanej bazie oraz pozwala na usystematyzowanie związków między danymi.

3.1. Definicja zbiorów encji określonych w projekcie

Wymagany zbiór encji najłatwiej uzyskać poprzez wyodrębnienie rzeczowników z opisu projektu. Na podstawie analizy założeń projektowych założyliśmy następujący zbiór encji:

— Użytkownik
— Mecz
— Zakład
— Wynik
— Zakład długoterminowy
— Turniej
— Drużyna
— Stadion
— Trener
— Sędzia
— Zawodnik

3.2. Ustalenie związków i ich typów między encjami

Relacje biorące udział		Typ	Obowiązkowość		Stopień
Użytkownik	Zakład	1:n	Obowiązkowy	Opcjonalny	binarny
Użytkownik	ZakładD[1]	1:n	Obowiązkowy	Opcjonalny	binarny
Użytkownik	Turniej	n:m	Opcjonalny	Opcjonalny	binarny
Mecz	Zakład	1:n	Obowiązkowy	Opcjonalny	binarny
Turniej	ZakładD[1]	1:n	Obowiązkowy	Opcjonalny	binarny
Turniej	Mecz	1:n	Obowiązkowy	Opcjonalny	binarny
Sędzia	Mecz	1:n	Obowiązkowy	Opcjonalny	binarny
Stadion	Mecz	1:n	Obowiązkowy	Opcjonalny	binarny
Wynik	Mecz	1:1	Opcjonalny	Obowiązkowy	binarny
Drużyna	ZakładD[1]	1:1	Obowiązkowy	Obowiązkowy	binarny
Drużyna	Mecz	2:n	Obowiązkowy	Opcjonalny	binarny
Turniej	Drużyna	n:m	Opcjonalny	Opcjonalny	binarny
Turniej	Drużyna	1:n	Opcjonalny	Opcjonalny	binarny
Zawodnik	Drużyna	n:m	Obowiązkowy	Opcjonalny	binarny
Trener	Drużyna	n:m	Opcjonalny	Obowiązkowy	binarny

[1] skrót od ZakładDługoterminowy

Tabela 3.1. Typy wszystkich związków w schemacie

Użytkownik - Zakład

Związek użytkownika z zakładem odwzorowuje następującą zależność. Użytkownik stawia zakład. Każdy zakład ma swojego jednego właściciela którym jest dany użytkownik, natomiast jeden użytkownik może stawiać od zera do wielu zakładów.

Użytkownik - ZakładDługoterminowy

Związek użytkownika z zakładem długoterminowym jest przedstawieniem w systemie zdarzenia postawienia przez użytkownika zakładu długoterminowego. Użytkownik tworzy zakład długoterminowy,w ogólności może ich tworzyć wiele (może też nie stworzyć żadnego). Każdy zakład długoterminowy ma obowiązkowo swojego jedynego właściciela.

Użytkownik - Turniej

Związek pomiędzy użytkownikiem a turniejem symbolizuje zapis użytkownika na turniej. Na jeden turniej może być zapisanych od zera do wielu użytkowników. Każdy użytkownik może zapisać się od zera do wielu turniejów.

Mecz - Zakład

Kolejny związek jest reprezentacją fizycznego połączenia meczu i zakładu. Jeden zakład przewiduje wynik tylko jednego meczu. Na jeden mecz może być postawionych kilka zakładów, w szczególności zero.

Turniej - ZakładDługoterminowy

Każdy zakład długoterminowy przewiduje wynik dokładnie jednego turnieju. Na jeden turniej może zostać postawionych od zera do wielu zakładów długoterminowych.

Turniej - Mecz

W ramach jednego turnieju rozgrywane jest od zera (sytuacja raczej dziwna, jednak możliwa zanim statystyk doda do turnieju drużyny oraz spotkania) do wielu spotkań. Jeden mecz jest rozgrywany w ramach jednego turnieju, przy czym zgodnie z założeniem wykluczamy możliwość obstawiania spotkań towarzyskich czyli takich które nie są rozgrywane w ramach żadnego turnieju.

Sędzia - Mecz

Reprezentuje sędziowanie meczu przez sędziego. Jeden sędzia może sędziować wiele spotkań (oczywiście nie równocześnie), natomiast jedno spotkanie ma tylko jednego sędziego głównego.

Stadion - Mecz

Podobnie jak w poprzednim związku, jeden mecz jest rozgrywany na jednym stadionie. Jeden stadion może być gospodarzem wielu spotkań.

Wynik - Mecz

Każdy mecz kończy się jakimś wynikiem. Zdecydowaliśmy się na rozwiązanie w którym wynik meczu jest przechowywany w osobnej encji. Jest to związek 1:1, jeden wynik obowiązkowo odpowiada jednemu spotkaniu. Warto zwrócić uwagę na opcjonalność związku ze strony encji meczu. Mecz który się jeszcze nie zakończył, nie może posiadać wyniku (system nie przewiduje obstawiania ustawionych spotkań).

Drużyna - Zakład Długoterminowy

Reprezentuje przewidywany wynik zakładu długoterminowego. Jeden zakład długoterminowy przewiduje jednego zwycięzce danego turnieju. Dana drużyna może wielokrotnie zostać wybrana przez użytkownika jako potencjalny zwycięzca, choć wcale nie musi być wybrana kiedykolwiek.

Drużyna - Mecz

Związek reprezentuje uczestnictwo drużyn w meczu. Każdy mecz musi mieć swoich uczestników. Zgodnie z założeniem, system pozwala na obstawianie spotkań pomiędzy dwoma drużynami, nie będą obsługiwane dyscyplinu typu skoki narciarskie gdzie liczba uczestników jest różna od dwóch. Jedna drużyna może grać wiele spotkań, w szczególności żadnego.

Turniej- Drużyna

W ramach jednego turnieju, gra w nim tylko określony zbiór drużyn. Zgodnie z założeniem, drużyny mogą być dynamicznie dodawane do turnieju dlatego też zakładamy że turniej może mieć zero uczestniczących drużyn. Z drugiej strony, jedna drużyna może brać udział w wielu turniejach czy chociażby w różnych edycjach tych samych rozgrywek.

Turniej - Drużyna *ponownie...*

Turnieje i drużyny łączy jeszcze jeden związek, który zdecydowaliśmy się specjalnie wyróżnić. Drugi związek pomiędzy turniejem a drużyną, reprezentuje zwycięzcę turnieju. W przypadku zwycięzców turniejów, zdecydowaliśmy się na przedstawienie tego nie za pomocą kolejnej encji (tak jak w przypadku encji *Wynik*), tylko przy pomocy związku. Jeden turniej ma jednego zwycięzce, który zostaje poznany po jego zakończeniu. Jedna drużyna może wygrywać kilka turniejów, w szczególności nie musi wygrać jakiegokolwiek.

Zawodnik - Drużyna

Jedna drużyna w danym momencie może mieć zakontraktowanych wielu zawodników oraz dodatkowo mieć historię zakończonych kontraktów z jeszcze większą ilością sportowców. Z drugiej strony, jeden zawodnik w swojej karierze może być związany kontraktem z wieloma drużynami.

Trener - Drużyna

Podobnie jak w przypadku poprzedniego punktu, związek *Trener - Drużyna* reprezentuje kontrakty trenerów z drużynami. Jedna drużyna ma (zazwyczaj) jednego głównego trenera, jednak ma przeszłość z innymi szkoleniowcami. Z drugiej strony, jeden szkoleniowiec może w swoim życiu być związany z kilkoma (rekordziści nawet z kilkunastoma) drużynami.

3.3. Określenie atrybutów i ich dziedzin

Użytkownik

Nazwa atrybutu	Typ i dziedzina	Obowiązkowy?	Opis
UżytkownikId	liczbowy	TAK	Numer identyfikujący użytkownika
Login	napisowy	TAK	Login do systemu oraz nazwa pod którą użytkownik będzie widoczny w systemie
Hasło	napisowy	TAK	Hash hasła, którym użytkownik będzie logował się do systemu
Salt	napisowy	TAK	Ciąg znaków zaburzający hasło, celem utrudnienia jego złamania
Imię	napisowy	TAK	Imię użytkownika, wymagane do identyfikacji i ewentualnego przekazania nagrody
Nazwisko	napisowy	TAK	Nazwisko użytkownika, wymagane do identyfikacji i ewentualnego przekazania nagrody

Tabela 3.2. Atrybuty encji Użytkownik

Mecz

Nazwa atrybutu	Typ i dziedzina	Obowiązkowy?	Opis
MeczId	liczbowy	TAK	Numer identyfikujący mecz
DataRozpoczęcia	datowy	TAK	Dzień i godzina rozpoczęcia meczu, do tego momentu można dodawać zakłady.

Tabela 3.3. Atrybuty encji Mecz

Zakład

Nazwa atrybutu	Typ i dziedzina	Obowiązkowy?	Opis
ZakładId	liczbowy	TAK	Numer identyfikujący zakładu
WynikGospodarzy	liczbowy	TAK	Przewidywany wynik zdobyty przez drużynę gospodarzy
WynikGości	liczbowy	TAK	Przewidywany wynik zdobyty przez drużynę gości
DataZawarcia	datowy	TAK	Data i godzina dodania/ostatniej edycji zakładu

Tabela 3.4. Atrybuty encji Zakład

Wynik

Nazwa atrybutu	Typ i dziedzina	Obowiązkowy?	Opis
WynikId	liczbowy	TAK	Numer identyfikujący wynik
WynikGospodarzy	liczbowy	TAK	Faktyczny wynik zdobyty przez drużynę gospodarzy
WynikGości	liczbowy	TAK	Faktyczny wynik zdobyty przez drużynę gości
DataDodania	datowy	TAK	Data i godzina dodania wyniku, potrzebna przy ewentualnej weryfikacji

Tabela 3.5. Atrybuty encji Wynik

Zakład Długoterminowy

Nazwa atrybutu	Typ i dziedzina	Obowiązkowy?	Opis
ZakładDługoterminowyId	liczbowy	TAK	Numer identyfikujący zakład długoterminowy
DataZawarcia	datowy	TAK	Data i godzina dodania/ostatniej edycji zakładu.

Tabela 3.6. Atrybuty encji Zakład Długoterminowy

Turniej

Nazwa atrybutu	Typ i dziedzina	Obowiązkowy?	Opis
TurniejId	liczbowy	TAK	Numer identyfikujący turniej
Nazwa	napisowy	TAK	Pełna nazwa turnieju
DataRozpoczęcia	datowy	TAK	Data i godzina rozpoczęcia turnieju, tj. pierwszego spotkania w tym turnieju.

Tabela 3.7. Atrybuty encji Turniej

Drużyna

Nazwa atrybutu	Typ i dziedzina	Obowiązkowy?	Opis
DrużynaId	liczbowy	TAK	Numer identyfikujący drużynę
Nazwa	napisowy	TAK	Pełna nazwa drużyny
Miasto	napisowy	TAK	Miasto z którego pochodzi drużyna
Narodowość	napisowy	TAK	Kraj z którego pochodzi drużyna

Tabela 3.8. Atrybuty encji Drużyna

Stadion

Nazwa atrybutu	Typ i dziedzina	Obowiązkowy?	Opis
StadionId	liczbowy	TAK	Numer identyfikujący stadion
Nazwa	napisowy	TAK	Pełna nazwa areny
Pojemność	liczbowy	TAK	Pojemność widowni
Miasto	napisowy	TAK	Miasto w którym znajduje się stadion
Narodowość	napisowy	TAK	Nazwa ulicy przy której znajduje się stadion

Tabela 3.9. Atrybuty encji Stadion

Trener

Nazwa atrybutu	Typ i dziedzina	Obowiązkowy?	Opis
TrenerId	liczbowy	TAK	Numer identyfikujący trenera
Imię	napisowy	TAK	Imię/imiona trenera
Nazwisko	liczbowy	TAK	Nazwisko trenera
DataUrodzenia	datowy	TAK	Data urodzenia trenera
Narodowość	napisowy	TAK	Kraj pochodzenia

Tabela 3.10. Atrybuty encji Trener

Sędzia

Nazwa atrybutu	Typ i dziedzina	Obowiązkowy?	Opis
SędziaId	liczbowy	TAK	Numer identyfikujący sędziego
Imię	napisowy	TAK	Imię/imiona sędziego
Nazwisko	liczbowy	TAK	Nazwisko sędziego
DataUrodzenia	datowy	TAK	Data urodzenia sędziego
Narodowość	napisowy	TAK	Kraj pochodzenia

Tabela 3.11. Atrybuty encji Sędzia

Zawodnik

Nazwa atrybutu	Typ i dziedzina	Obowiązkowy?	Opis
SędziaId	liczbowy	TAK	Numer identyfikujący sędziego
Imię	napisowy	TAK	Imię/imiona sędziego
Nazwisko	liczbowy	TAK	Nazwisko sędziego
DataUrodzenia	datowy	TAK	Data urodzenia sędziego
Narodowość	napisowy	TAK	Kraj pochodzenia
Numer	liczbowy	TAK	Numer na koszulce

Tabela 3.12. Atrybuty encji Sędzia

3.4. Dodatkowe reguły integralnościowe

Na poziomie konceptualnym, wszystkie atrybuty zostały oznaczone jako `NOT NULL`. Dodatkowo, atrybut *Login* w encji *Użytkownik* musi być unikalny względem reszty danych w bazie, dlatego też został on opatrzony słowem `UNIQUE`.

Należy zapewnić aby w systemie nie doszło do sytuacji że zwycięzcą danego turnieju jest drużyna która nie jest jego uczestnikiem. Zgodnie podjęliśmy decyzję projektową że to zabezpieczenie zostanie zrealizowane po stronie aplikacji a nie po stronie bazy danych.

3.5. Klucze kandydujące i główne

Dla każdej encji spróbowaliśmy znaleźć unikalny klucz. Wyniki naszych starań znajdują się w tabeli poniżej.

Relacja	Klucz kandydujący
Użytkownik	Login
Mecz	—
Drużyna	Nazwa
Zawodnik	Imię & Nazwisko
Stadion	Adres & Nazwa
Zakład	—
Wynik	—
Turniej	Nazwa
Sędzia	Imię & Nazwisko
Trener	Imię & Nazwisko
ZakładDługoterminowy	—

Tabela 3.13. Proponowane klucze kandydujące

W tabeli znajdują się niepoprawne klucze takie jak kombinację imienia i nazwiska, które należy z miejsca odrzucić, pomimo faktu że prawdopodobieństwo istnienia dwóch zawodników o tym samym imieniu i nazwisku jest niezwykle niskie (niskie nie znaczy zerowe). Dodatkowo, doszliśmy do wniosku że nazwy turniejów z góry nie muszą być unikalne. To samo tyczy się nazw drużyn, które mogą prowadzić różne sekcje o tej samej nazwie. Po kolei odrzucając kolejne klucze, zdecydowaliśmy że jedynym sensownym kluczem kandydującym jest *Login* w encji *Użytkownik*, który musi być unikalny.

Z tego też powodu podjęliśmy kolejną decyzją projektową jaką było wstawienie do każdej encji sztucznego klucza głównego w postaci liczbowego identyfikatora.

Relacja	Klucz główny	Klucz kandydujący
Użytkownik	UżytkownikId	Login
Mecz	MeczId	—
Drużyna	DrużynaId	—
Zawodnik	ZawodnikId	—
Stadion	StadionId	—
Zakład	ZakładId	—
Wynik	WynikId	—
Turniej	TurniejId	—
Sędzia	SędziaId	—
Trener	TrenerId	—
ZakładDługoterminowy	ZakładDługoterminowyId	—

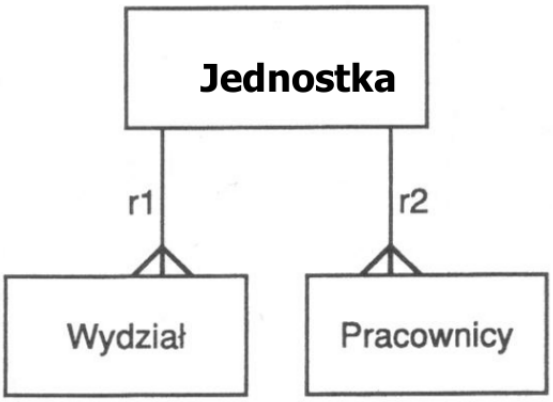
Tabela 3.14. Klucze główne i pozostałe kandydujące

3.6. Problem pułapek szczelinowych i wachlarzowych – analiza i przykłady

Przed przejściem do kolejnego etapu projektowania, należało rozważyć czy w naszym projekcie nie ukryły się żadne pułapki szczelinowe oraz wachlarzowe.

3.6.1. Pułapki wachlarzowe

Pułapką wachlarzową nazywamy sytuację w której nie ma jednoznacznego połączenia pomiędzy dwoma encjami które są w związku n:1 z tą samą encją. Najłatwiej ją zidentyfikować gdy z jednej encji wychodzi więcej niż jeden związek 1:n.



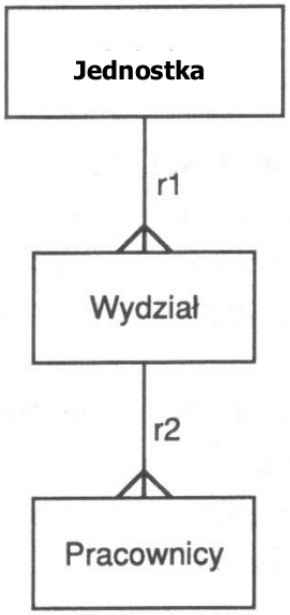
Rysunek 3.1. Przykład pułapki wachlarzowej przedstawionej na wykładzie[2]

Na przykładzie przedstawionym powyżej, nie jest możliwe jednoznaczne wywnioskowanie w którym wydziale pracuje dany pracownik, mimo iż znamy do której należy jednostki.

W początkowym projekcie systemu, do schematu wkradła nam się pułapka wachlarzowa pomiędzy encjami *Użytkownik*, *Turniej* oraz *Zakład*. Jej wynikiem był brak możliwości zapytania o wszystkie zakłady jednego użytkownika w ramach jednego turnieju. Pułapkę zlikwidowaliśmy poprzez dodanie związku pomiędzy encjami *Turniej* oraz *Zakład*. Po takiej modyfikacji zakład przechowuje informację w ramach jakiego turnieju został on stworzony co umożliwia zliczanie punktów.

3.6.2. Pułapki szczelinowe

Pułapką szczelinową nazywamy sytuację w której nie ma jednoznacznego połączenia pomiędzy encjami, połączonych związkami 1:n w następujący sposób.



Rysunek 3.2. Przykład pułapki szczelinowej przedstawionej na wykładzie[2]

Niepoprawnym jest założenie że zawsze znajdzie się połączenie po środku, brak połączenia nazywamy szczeliną. Rozwiązaniem tej sytuacji jest dodanie kolejnego związku między skrajnymi encjami.

W przedstawionym projekcie, po wnikliwej analizie stwierdziliśmy brak sensownych pułapek szczelinowych, tj. istnieją takie połączenia spełniające wymagania pułapki szczelinowej ale z punktu biznesowego takie połączenia nie mają sensu, dlatego też zostają one świadomie pominięte.

3.7. Schemat ER na poziomie konceptualnym

Schemat ER na poziome konceptualnym znajduje się w załączniku A dołączonym na końcu sprawozdania.

4. Model logiczny

4.1. Charakterystyka modelu relacyjnego

Model relacyjny został wprowadzony przez pracownika IBM, Edgara Franka Codda w 1970 roku[3]. Model ten opisuje dane za pomocą tabel nazywanych relacjami, które przechowują atomowe dane. Na relacjach są dobrze zdefiniowane operacje wyszukiwania, dodawania danych i ich edycji. Dodatkowo, w łatwy sposób definiowane są warunki spójności: zależności kluczowe i zawierania oraz ograniczenia na wartości danych poprzez typy czy dopuszczalne zbiory wartości.

Daleko posunięta niezależność danych zezwala na modyfikacje wewnętrznej reprezentacji danych, uporządkowania rekordów czystruktury plików bez wpływu na aplikacje wykorzystujące dane.

4.2. Usunięcie właściwości niekompatybilnych z modelem relacyjnym

4.2.1. Tablice bridge'ujące

Najpoważniejszą niekompatybilnością logicznego modelu relacyjnego z modelem konceptualnym jest odwzorowanie związków wiele do wielu. W modelu relacyjnym są one niedopuszczalne. Rozwiązaniem problemu jest wprowadzanie tablic bridge'ujących. Pozwalają one na reprezentację związków wiele do wielu za pomocą kolumn odpowiadających kluczom głównym zbiorów encji biorących udział w związku[2].

W prezentowanym wcześniej modelu konceptualnym występowały cztery związki wiele do wielu:

— *Użytkownik - Turniej*
— *Drużyna - Turniej*
— *Drużyna - Zawodnik*
— *Drużyna - Trener*

Dla każdego związku, stworzona została odpowiadająca jej tablica:

Związek n:m	Bridge
Użytkownik - Turniej	*ZapisUżytkownika*
Drużyna - Turniej	*ZapisDrużyny*
Drużyna - Zawodnik	*KontraktZawodnika*
Drużyna - Trener	*KontraktTrenerski*

Tabela 4.1. Zrealizowana zamiana związków wiele do wielu na tablice bridge'ujące

Dodatkowo, do tablic *KontraktZawodnika* i *KontraktTrenerski* dodane zostały dodatkowe atrybuty *DataZawarcia* i *DataZakończenia*. Pozwala to na przechowywanie informacji o historii (z zachowaniem chronologii) oraz informacji o tym który kontrakt jest aktualnie ważny (jest to ten bez daty zakończenia).

4.2.2. Liczby mnogie

W związku z inną konwencją nazewniczą, zmieniliśmy nazwy relacji tak aby ich nazwy były w liczbie mnogiej.

Stara nazwa	Nowa nazwa
Użytkownik	Użytkownicy
Mecz	Mecze
Zakład	Zakłady
Wynik	Wyniki
ZakładDługoterminowy	ZakładyDługoterminowe
Turniej	Turnieje
Drużyna	Drużyny
Stadion	Stadiony
Trener	Trenerzy
Sędzia	Sędziowie
Zawodnik	Zawodnicy
ZapisUżytkownika	ZapisyUżytkowników
ZapisDrużyny	ZapisyDrużyn
KontraktZawodnika	KontraktyZawodników
KontraktTrenerski	KontraktyTrenerskie

Tabela 4.2. Zrealizowana zmiana nazw relacji

4.2.3. Typy danych

W modelu konceptualnym, określiliśmy typy danych w sposób bardzo ogólny. Na tym etapie dokonamy ich specyfikacji, typowi liczbowemu przypisaliśmy typ `INTEGER`, typowi napisowemu przyporządkowaliśmy typ `VARCHAR` a typowi datowemu typ `DATE`. Długość typów napisowych została dostosowana do potencjalnych wartości opisywanego atrybutu.

4.3. Proces normalizacji

Normalizacja bazy polega na eliminacji powtarzających się (redundantych) wpisów. W bazie znormalizowanej, dane nie są w żaden sposób powtarzane a tylko i wyłącznie ze sobą linkowane. Zwiększa to bezpieczeństwo i zmniejsza ryzyko powstania niespójności.

Normalizacja polega tylko i wyłącznie na przekształcaniu schematu bazy danych. Rezultatem procesu normalizacji nigdy nie jest utrata danych, ewentualnie ich przyrost poprzez dodanie nowych kluczy[2].

4.3.1. Pierwsza postać normalna

Przytoczmy definicję pierwszej postaci normalnej:

Relacja jest w pierwszej postaci normalnej, jeśli każda wartość atrybutu w każdej krotce tej relacji jest wartością elementarną, czyli nierozkładalną oraz jeśli nie ma powtarzających się grup[2].

W myśl tej definicji, zlokalizowaliśmy dwie relację które na pierwszy rzut oka mogą nie spełniać warunków pierwszej postaci normalnej.

Relacja *Stadiony*

Atrybut *Adres* relacji *Stadiony* nie jest polem atomowym. Adres składa się z kilku elementów, takich jak miasto, ulica, numer ulicy i kod pocztowy. W celu doprowadzenia relacji do 1PN, atrybut ten został rozbity na 4 wcześniej wymienione.

Relacja *Wyniki*

Relacja *Wyniki* potencjalnie może przechowywać duże grupy bardzo podobnych rekordów, różniących się stemplem czasowym. Zdecydowaliśmy się na wprowadzenie dodatkowej relacji *SłownikWyników*, która przechowuje powtarzające się wyniki.

Atrybuty *Narodowość*

W wielu relacjach występują atrybuty określające narodowość. Podobnie jak w przypadku wyników, jest to miejsce na potencjalne powtarzające się grupy danych, dlatego też zdecydowaliśmy się na stworzenie dodatkowej relacji przechowującej informację o narodowości.

4.3.2. Druga postać normalna

Przypomnijmy definicję drugiej postaci normalnej:

Relacja jest w drugiej postaci normalnej, jeśli jest w 1PN oraz każdy atrybut tej relacji nie wchodzący w skład żadnego klucza potencjalnego jest w pełni funkcyjnie zależny od wszystkich kluczy potencjalnych tej relacji.[2].

Z definicji tej można wyciągnąć wniosek że jeżeli relacja jest w pierwszej postaci normalnej i ma wszystkie klucze potencjalne proste to jest automatycznie w drugiej postaci normalnej.

We wszystkich (oprócz jednej) relacjach zastosowany jest prosty klucz potencjalny, co automatycznie załatwią sprawę spełniania 2PN.

Relacja *Użytkownicy*

Jedyną podejrzaną relacją jest relacja przechowująca dane o użytkownikach. Aby zachować spójność dla developera aplikacji, zdecydowaliśmy się we wcześniejszym kroku na wstawienie sztucznego identyfikatora, pomimo unikalnego loginu każdego użytkownika. Nie ma jednak tutaj mowy o żadnych zależnościach funkcyjnych między pozostałymi atrybutami krotki.

4.3.3. Trzecia postać normalna

Trzecia postać normalna ma następującą definicję:

Relacja jest w trzeciej postaci normalnej, jeśli jest ona w drugiej postaci normalnej i każdy jej atrybut nie wchodzący w skład żadnego klucza potencjalnego nie jest przechodnio funkcyjnie zależny od żadnego klucza potencjalnego tej relacji.[2].

Z definicji bezpośrednio wynika że wszystkie niekluczowe kolumny zależą tylko od całości klucza. Ponownie, jak w przypadku rozważań na temat drugiej postaci normalnej, mamy tylko jedną relację podejrzaną o niespełnianie definicji 3PN.

Relacja *Użytkownicy* ponownie...

Wcielmy się w rolę hipotetycznego użytkownika aplikacji opartej na projektowanej bazie danych. Do logowania się do systemu podaje ona tylko i wyłącznie swój login. Na jego podstawie znajdowany jest hash hasła oraz zabezpieczający

ciąg zaburzający i porównywany z dostarczonym hasłem. W całej tej historii ani razu nie został użyty sztuczny identyfikator. Odpowiednia atrybuty krotki zostały zlokalizowane za pomocą niepełnego klucza. W związku z tym, aby spełnić postulaty 3PN, pozbyliśmy się sztucznego identyfikatora *UżytkownikId* z relacji *Użytkownicy*, w wyniku czego prostym kluczem głównym stał się atrybut *Login*.

4.4. Proces denormalizacji

Denormalizacja jest procesem jawnego odejścia od założonej postaci normalnej, celem optymalizacji pracy z bazą. W ramach tego procesu, częściowo cofnęliśmy część zmian wprowadzonych przy normalizacji bazy.

Relacja *Użytkownicy* po raz trzeci...

W ramach zachowania spójności z resztą relacji, przywróciliśmy klucz główny *UżytkownikId* tak aby programista aplikacji nie musiał zagłębiać się w rozbudowaną dokumentację prezentowanej bazy danych. Atrybut *Login* został cofnięty do swojej pierwotnej roli, wymuszona na nim została unikalność, jednak do definiowania związków między relacjami przekazywany będzie numeryczny identyfikator.

Relacja *Stadiony* (ponownie?)

Przechowywanie całego adresu pocztowego stadionu wydaje się dość bezsensownym pomysłem, biorąc pod uwagę przeznaczenie bazy. Najważniejszym z całego adresu jest nazwa ulicy, od której najczęściej pochodzi nazwa lub chociaż przydomek stadionu (przykładowo każdy fan Liverpool'u wie że ich stadion znajduje się przy Anfield Road, jednak mało który zna kod pocztowy tego miejsca). W związku z tym, zdecydowaliśmy się na pozostawienie atrybutu *Miasto* oraz złączenie pozostałych atrybutów adresowych w jeden *Adres*, który nie będzie przechowywał informacji pocztowych a jedynie te pozwalające na wizualną identyfikację położenia stadionu.

Relacja *Wyniki* (również ponownie)

Wyodrębnienie relacji *SłownikWyników* na pierwszy rzut oka wydaje się doskonałym pomysłem, jednak dodaje dodatkowy narzut obliczeniowy na silnik bazy danych. Duplikacja wyniku spotkania (dwóch liczb!) nie wydaje się taką niedogodnością jaką jest znacznie dłuższy czas dostępu do danych, dlatego też zdecydowaliśmy się na wycofanie się z wprowadzonych zmian i powrót do pierwotnej wersji gdzie pola *WynikGospodarzy* oraz *WynikGości* nie są wyodrębnione do osobnej encji tylko są atrybutami encji *Wynik*.

4.5. Schemat ER na poziomie modelu logicznego

Schemat ER na poziome modelu logicznego znajduje się w załączniku B dołączonym na końcu sprawozdania.

5. Faza fizyczna

5.1. Weryfikacja wykonywalności transakcji

Transakcja	Potrzebne zasoby	Wykonalne?
Zapis na turniej	Turnieje, ZapisUżytkowników	TAK
Utworzenie zakładu	Zakłady, Mecze	TAK
Edycja zakładu	Zakłady	TAK
Utworzenie zakładu długoterminowego	ZakładyDługoterminowe, Drużyne, Turnieje	TAK
Edycja zakładu długoterminowego	ZakładyDługoterminowe, Drużyne	TAK
Podgląd tabeli wyników	Mecze, Turnieje, Zakłady, Użytkownicy, ZakładyDługoterminowe	TAK
Utworzenie turnieju	Turnieje	TAK
Zapis drużyny na turniej	Turnieje, drużyny	TAK
Dodanie meczu	Mecze, Stadiony, Sędziowie, Drużyny	TAK
Dodanie drużyny	Drużyny	TAK
Dodanie wyniku	Wyniki, Mecze	TAK
Dodanie stadionu	Stadiony	TAK
Dodanie sędziów	Sędziowie	TAK
Dodanie trenerów	Trenerzy	TAK
Dodanie zwycięzcy turnieju	Turnieje, Drużyny	TAK
Aktualizowanie składów drużyn	Drużyny, Zawodnicy, Trenerzy	TAK
Utworzenie konta	Użytkownicy	TAK
Zmiana hasła	Użytkownicy	TAK

Tabela 5.1. Wyniki zweryfikowania wykonywalności transakcji

5.2. Dobór indeksów

Indeks jest specjalną strukturą danych wprowadzoną w celu zwiększenia prędkości wykonywania operacji na tabeli. Wykorzystuje się je przy zapytaniach typu DQL (SELECT), które mają na celu wyszukiwanie odpowiednich wartości w bazie danych. Podczas realizacji zapytania optymalizator najpierw przeszukuje indeks, który jest uporządkowany, a następnie na podstawie indeksu odczytuje odpowiednie rekordy. Indeks posiada strukturę logiczną i fizyczną niezależną od tabeli, do jakiej się odwołuje. Posiada również własną przestrzeń dyskową oraz jest automatycznie utrzymywany przez system zarządzania bazą danych[4].

W celu poprawienia wydajności podczas pracy z bazą danych, zostały dobrane indeksy w miejscu, w którym wyszukiwania danych będą najczęstsze - w tabeli zakładów.

Ostatecznie założyliśmy dwa indeksy na kolumnach definiujących związki:
— MeczId
— UżytkownikId

5.3. Skrypt SQL zakładający bazę danych

Skrypt zakładający bazę danych znajduje się w załączniku C dołączonym na końcu sprawozdania.

5.4. Przykłady zapytań i poleceń SQL odnoszących się do bazy danych

```
INSERT INTO Drużyny VALUES (10, 'Real Madrid', 'Madryt', 6);

INSERT INTO Uzytkownicy VALUES (111, 'BeastMaster64', 'dabkhsfbahfbia',
'ad5sdacagj613', ' Felix', 'Khjellberg');

INSERT INTO Sedziowie (SędziaId, Imię, Nazwisko, Narodowość) VALUES
(1, 'Howard', 'Webb', 3);

SELECT DataZawarcia FROM Zaklady WHERE ZakladId = 10;

SELECT Pojemnosc FROM Stadiony WHERE Nazwa = 'Estadio Santiago Bernabeu';

SELECT * FROM Uzytkownicy WHERE Imię LIKE 'J%';

SELECT Imię, Nazwisko, Numer FROM Zawodnicy ORDER BY Numer ASC;
```

Bibliografia

[1] Robert Wrembel, *Wykład z przedmiotu Bazy Danych. Wykład 3: Modelowanie danych, Model związków-encji.*, Poznań, 2006.

[2] Marcin Kowalczyk, *Wykład z przedmiotu Wprowadzanie do Baz Danych.*, Warszawa, 2018.

[3] Edgar Frank Codd, *A relational model of data for large shared data banks.* Commun. ACM, New York, 1970.

[4] Jacek Włodarski, *SQL Server – Indeksy – kiedy i jak je stosować?.* MSDN@Microsoft, 2011.

[5] Worldwide Narrow Vision Gang, *Rewers.* UNDA Records, Gdynia, 2018.

A. Schemat ER na poziomie konceptualnym

B. Schemat ER na poziomie modelu logicznego

C. Skrypt zakładający bazę danych