Bazy danych i Big Data (BDBT)

Projekt – część I

Stadnina

Radosław Pietkun

Grupa B

Semestr 20Z

Spis treści

1. Zakres i cel projektu

Celem projektu jest zaprojektowanie i zaimplementowanie relacyjnej bazy danych, której zadaniem będzie wspieranie działalności stadniny.

Stadnina zajmuje się hodowlą i sprzedażą koni. Posiada bazę zwierząt, pracowników i obiektów należących do infrastruktury przedsiębiorstwa (takich jak stajnie, place, hale). Projektowana baza musi również uwzględniać, jakie są obowiązki danego pracownika (za jakie zwierzęta oraz obiekty jest odpowiedzialny) oraz które zwierzęta wykorzystują dany obiekt. Należy również przechowywać informacje o wypłacanych pracownikom wynagrodzeniach. Wśród pracowników można wyszczególnić kilka specjalizacji związanych z wymaganymi kwalifikacjami na dane stanowisko (weterynarz, instruktor jazdy konnej). Przedsiębiorstwo prowadzi sprzedaż koni przez publikowanie ofert sprzedaży. Przechowuje również dane o swoich klientach oraz o przeprowadzonych transakcjach.

2. Definicja systemu

2.1. Perspektywy użytkowników

- Kierownik stadniny – osoba mająca dostęp do wszystkich danych w bazie.
- Księgowy – osoba odpowiedzialna za prowadzenie finansów stadniny, ma dostęp do ofert sprzedaży, transakcji zawartych z klientami, wynagrodzeń pracowników.
- Pracownik biurowy – osoba pracująca w biurze stadniny, zajmuje się zarządzaniem ofertami sprzedaży zwierząt, prowadzeniem ewidencji zwierząt, obiektów, pracowników.
- Pracownik fizyczny – osoba pracująca na terenie stadniny, ma dostęp do swoich danych osobowych, danych o swoich wynagrodzeniach oraz o swoich obowiązkach.

2.2. Transakcje (operacje na danych)

Poniższa tabela przedstawia główne transakcje realizowane w stadninie z uwzględnieniem, kto ma uprawnienia do przeprowadzenia poszczególnych operacji.

Transakcje	**Kierownik**	**Księgowy**	**Pracownik biurowy**	**Pracownik fizyczny**
Obsługa pracowników				
Dodanie/usunięcie pracownika z bazy	T	N	T	N
Modyfikacja danych pracownika	T	N	T	T
Wprowadzenie danych o wynagrodzeniu	T	T	N	N
Przypisanie pracownika do danego zadania	T	N	T	N
Przeglądanie informacji o pracownikach	T	T	T	TYLKO O SOBIE

Obszar sprzedaży				
Tworzenie nowych/usuwanie ofert sprzedaży	T	N	T	N
Modyfikowanie oferty	T	N	T	N
Obsługa sprzedanych ofert	T	T	N	N
Obsługa obiektów				
Dodanie nowego/usunięcie obiektu z bazy	T	N	T	N
Modyfikacja danych o obiekcie	T	N	T	N
Przeglądanie informacji o obiektach	T	T	T	T
Obsługa zwierząt				
Dodanie/usunięcie zwierzęcia z bazy	T	N	T	N
Modyfikacja danych o zwierzęciu	T	N	T	N
Przypisanie zwierzęcia do danego obiektu	T	N	T	N
Przeglądanie informacji o zwierzętach	T	T	T	T
Obsługa klientów				
Dodanie klienta do bazy	T	N	T	N
Modyfikacja danych klienta	T	N	T	N
Przypisanie klienta do danej transakcji	T	T	N	N
Przeglądanie informacji o klientach	T	T	T	N
Zarządzanie stadniną				
Modyfikacja danych o stadninie	T	N	N	N
Dodanie nowej/likwidacja starej stadniny z bazy	T	N	N	N

3. Model konceptualny

3.1. Definicje zbiorów encji określonych w projekcie

- ***Stadnina*** – przedsiębiorstwo, zajmujące się hodowlą i sprzedażą koni, dla którego projektowana jest baza danych.
- ***Klient*** – klient stadniny, osoba kupująca wybraną ofertę sprzedaży związaną z danym koniem.
- ***Koń*** – zwierzę należące do stadniny, hodowane w celu późniejszej sprzedaży klientowi.
- ***Obiekt*** – element należący do infrastruktury stadniny i użytkowany przez zwierzęta i/lub przez pracowników (np. stajnie, hale, place, wybiegi).
- ***Oferta sprzedaży*** – oferta dotycząca konkretnego konia, proponowana klientom do nabycia.
- ***Pracownik*** – pracownik stadniny, zarówno osoba pracująca w biurze jak i fizycznie na terenie stadniny, można wyróżnić dwie specjalizacje (***weterynarz*** i ***instruktor***), które związane są z dodatkowymi wymaganiami potwierdzającymi kwalifikacje pracownika do wykonywania zawodu.

Encje

Nazwa encji	**Klucz główny (PUI)**	**Liczba atrybutów**
Stadnina	Nr_stadniny	10
Klient	Nr_klienta	9
Kon	Nr_konia	13
Obiekt	Nr_obiektu	5
Oferta_sprzedazy	Nr_oferty	3
Pracownik	Nr_pracownika	13
Instruktor		5
Weterynarz		4

Dziedziczenie

Nazwa dziedziczenia	Specjalizacja_pracownika
Encja – rodzic	Pracownik
Encje – dzieci	Instruktor, Weterynarz
Sposób implementacji	n tabel

3.2. Związki między encjami i ich typy

- *Stadnina **posiada** konia* – stadnina może posiadać wiele koni lub zero, jeśli dopiero zaczyna działalność; koń może należeć do dokładnie jednej stadniny.
- *Stadnina **zatrudnia** pracownika* – stadnina może zatrudniać wielu pracowników lub zero, jeśli dopiero zaczyna działalność; pracownik może pracować w dokładnie jednej stadninie.
- *Stadnina **ma w posiadaniu** obiekt* – stadnina może posiadać wiele obiektów lub zero, jeśli dopiero zaczyna działalność; obiekt może należeć do dokładnie jednej stadniny.
- *Koń **korzysta z** obiektu* – koń może być przypisany do wielu obiektów (przynajmniej jednego, w którym będzie spał); obiekt może być wykorzystywany przez wiele koni (lub chwilowo przez żadnego).
- *Pracownik **nadzoruje** obiekt* – pracownik nadzoruje wiele obiektów lub zero, jeśli jest to pracownik biurowy; obiekt może być nadzorowany przez wielu pracowników (przynajmniej przez jednego). Nadzór polega np. na czyszczeniu obiektu lub wykonywaniu prac gospodarczych na jego terenie.
- *Pracownik **opiekuje się** koniem* – pracownik jest odpowiedzialny za wiele koni lub za żadnego, jeśli jest to pracownik biurowy; koń może być pod opieką wielu pracowników. Opieka polega m.in. na karmieniu, pojeniu oraz czyszczeniu koni.
- *Oferta sprzedaży **dotyczy** konia* – oferta sprzedaży dotyczy dokładnie jednego konia; koń może być w jednej ofercie lub w żadnej, jeśli nie jest obecnie na sprzedaż.
- *Klient **kupuje** ofertę* – klient może kupić wiele ofert lub żadną; oferta może być kupiona przez jednego klienta lub w ogóle.

Nazwa związku	Rodzaj związku	Encje wchodzące w związek i typ uczestnictwa	Typ i krotności związku	Stopień związku
posiada_konia	Niezidentyfikowany	Stadnina – obowiązkowy Kon – opcjonalny	1..1 - 0..m	Binarny
zatrudnia	Niezidentyfikowany	Stadnina – obowiązkowy Pracownik – opcjonalny	1..1 - 0..m	Binarny
ma_w_posiadaniu	Niezidentyfikowany	Stadnina – obowiązkowy Obiekt – opcjonalny	1..1 - 0..m	Binarny
korzysta	Niezidentyfikowany	Kon – opcjonalny Obiekt – obowiązkowy	0..n - 1..m	Binarny
nadzoruje	Niezidentyfikowany	Pracownik – obowiązkowy Obiekt – opcjonalny	1..n - 0..m	Binarny
opiekuje_sie	Niezidentyfikowany	Pracownik – obowiązkowy Kon – opcjonalny	1..n - 0..m	Binarny
dotyczy	Niezidentyfikowany	Oferta_sprzedazy – opcjonalny Kon – obowiązkowy	0..1 - 1..1	Binarny
kupuje_oferte	Niezidentyfikowany	Klient – opcjonalny Oferta_sprzedazy – opcjonalny	0..1 - 0..m	Binarny

3.3. Atrybuty i ich dziedziny

Atrybuty encji 'Stadnina'

PUI	Nazwa atrybutu	Dziedzina	Typ danych	Czy pole obowiązkowe	Czy atrybut prosty	Opis
T	Nr_stadniny		SmallInt	T	T	Unikatowy numer stadniny
N	Nazwa		VarChar(30)	T	T	Nazwa stadniny
N	Data_zalozenia		Date	T	T	Data założenia stadniny
N	Adres		VarChar(400)	T	N	Adres stadniny, pole segmentowe (miejscowość, ulica, numer budynku, kod pocztowy, poczta)
N	Wlasciciel		VarChar(400)	T	N	Właściciel stadniny, pole segmentowe (imię, nazwisko) i wielowartościowe (w przypadku kilku właścicieli)
N	Mail		VarChar(30)	T	T	Adres mailowy
N	Nr_telefonu		VarChar(12)	T	T	Numer telefonu
N	NIP		Character(10)	T	T	Numer NIP
N	REGON		Character(9)	T	T	Numer REGON
N	Strona_WWW		VarChar(30)	N	T	Strona internetowa

Atrybuty encji 'Klient'

PUI	Nazwa atrybutu	Dziedzina	Typ danych	Czy pole obowiązkowe	Czy atrybut prosty	Opis
T	Nr_klienta		Integer	T	T	Unikatowy numer klienta
N	Imie		VarChar(20)	T	T	Imię klienta
N	Nazwisko		VarChar(30)	T	T	Nazwisko klienta
N	Plec	PlecD	Character(1)	T	T	Płeć klienta [K, M]
N	Adres		VarChar(400)	N	N	Adres zamieszkania klienta (pole segmentowe)
N	Nr_telefonu		Character(12)	T	T	Numer telefonu
N	Mail		VarChar(30)	T	T	Adres mailowy klienta
N	NIP		Character(10)	N	T	Numer NIP
N	PESEL		Character(11)	N	T	Numer PESEL

Atrybuty encji 'Kon'

PUI	Nazwa atrybutu	Dziedzina	Typ danych	Czy pole obowiązkowe	Czy atrybut prosty	Opis
T	Nr_konia		Integer	T	T	Unikatowy numer konia
N	Nazwa		VarChar(30)	T	T	Nazwa konia
N	Plec_konia	PlecKoniaD	VarChar(10)	T	T	Płeć konia [OGIER, KLACZ, WALACH]
N	Masc		VarChar(30)	T	T	Maść (kolor) konia
N	Data_urodzenia		Date	T	T	Data urodzenia konia
N	Typ_konia	TypKoniaD	VarChar(20)	T	T	Typ konia [GORACOKRWISTY, ZIMNOKRWISTY, KUC]
N	Rasa		VarChar(60)	T	N	Rasa konia, pole segmentowe (kod i nazwa rasy)
N	Rod_meski		VarChar(30)	T	T	Ród męski
N	Linia_zenska		VarChar(30)	T	T	Linia żeńska
N	Wymiary		VarChar(20)	N	N	Wymiary konia, pole wielowartościowe (wysokość konia w kłębie [cm]. obwód klatki piersiowej [cm], obwód lewej nogi nad pęciną [cm]), pole nieobowiązkowe np. dla ciągle rosnących źrebiąt
N	Przeznaczenie		VarChar(20)	T	T	Przeznaczenie konia
N	Kraj_pochodzenia		VarChar(30)	T	T	Kraj pochodzenia konia
N	Dodatkowe_informacje		VarChar(800)	N	N	Dodatkowe informacje, opis i charakterystyka konia

Atrybuty encji 'Obiekt'

PUI	Nazwa atrybutu	Dziedzina	Typ danych	Czy pole obowiązkowe	Czy atrybut prosty	Opis
T	Nr_obiektu		Integer	T	T	Unikatowy numer obiektu
N	Nazwa		VarChar(30)	T	T	Nazwa obiektu
N	Powierzchnia		Bigint	T	T	Powierzchnia obiektu [m^2]
N	Opis		VarChar(800)	T	N	Opis obiektu
N	Czy_do_uzytku	CzyDoUzytkuD	Character(1)	T	T	Czy obiekt nadaje się do użytku [T, N]

Atrybuty encji 'Oferta_sprzedazy'

PUI	Nazwa atrybutu	Dziedzina	Typ danych	Czy pole obowiązkowe	Czy atrybut prosty	Opis
T	Nr_oferty		Integer	T	T	Unikatowy numer oferty
N	Status_oferty	StatusOfertyD	VarChar(20)	T	T	Status oferty sprzedaży [AKTYWNA, SPRZEDANA, NIEAKTYWNA]
N	Cena		Money	T	T	Cena sprzedaży

Atrybuty encji 'Pracownik'

PUI	Nazwa atrybutu	Dziedzina	Typ danych	Czy pole obowiązkowe	Czy atrybut prosty	Opis
T	Nr_pracownika		Integer	T	T	Unikatowy numer pracownika
N	Imie		VarChar(20)	T	T	Imię pracownika
N	Nazwisko		VarChar(30)	T	T	Nazwisko pracownika
N	Stanowisko		VarChar(30)	T	T	Stanowisko pracownika
N	Data_urodzenia		Date	T	T	Data urodzenia
N	PESEL		Character(11)	N	T	Numer PESEL
N	Plec	PlecD	Character(1)	T	T	Płeć pracownika [K, M]
N	Adres		VarChar(400)	T	N	Adres pracownika, pole segmentowe
N	Nr_telefonu		VarChar(12)	N	T	Numer telefonu
N	Mail		VarChar(30)	N	T	Adres mailowy
N	Wynagrodzenie		Money	N	N	Wynagrodzenie, pole wielowartościowe (kwoty wynagrodzeń miesięcznych)
N	Data_zatrudnienia		Date	T	T	Data zatrudnienia
N	Nr_konta		Character(26)	N	T	Numer konta, pole opcjonalne, bo niektórzy mogą pracować w charakterze wolontariatu

Atrybuty encji 'Instruktor' (potomek encji ’Pracownik’)

PUI	Nazwa atrybutu	Dziedzina	Typ danych	Czy pole obowiązkowe	Czy atrybut prosty	Opis
N	Stopien		VarChar(30)	T	T	Stopień jeździecki
N	Nr_licencji		VarChar(20)	T	T	Numer licencji jeździeckiej
N	Termin_waznosci_uprawnien		Date	T	T	Termin ważności uprawnień jeździeckich
N	Data_wydania_licencji		Date	T	T	Data wydania licencji jeździeckiej
N	Organ_wydajacy		VarChar(30)	T	T	Organ wydający licencję

Atrybuty encji 'Weterynarz' (potomek encji ’Pracownik’)

PUI	Nazwa atrybutu	Dziedzina	Typ danych	Czy pole obowiązkowe	Czy atrybut prosty	Opis
N	Nr_PWZ		Character(5)	T	T	Numer prawa wykonywania zawodu
N	Nazwa_okregowej_izby		VarChar(100)	T	T	Nazwa okręgowej izby lekarsko-weterynaryjnej
N	Dodatkowe_uprawnienia		VarChar(400)	N	N	Dodatkowe uprawnienia, pole wielowartościowe

Dziedziny zdefiniowane na potrzeby projektu

Nazwa dziedziny	Typ danych	Dopuszczalne wartości	Nazwa atrybutu	Nazwa encji
CzyDoUzytkuD	Character(1)	‘T’, ‘N’	Czy_do_uzytku	Obiekt
PlecD	Character(1)	‘K’, ‘M’	Plec	Pracownik, Klient
PlecKoniaD	VarChar(10)	‘OGIER’, ‘KLACZ’, ‘'WALACH'	Plec_konia	Kon
StatusOfertyD	VarChar(20)	'AKTYWNA', 'SPRZEDANA', 'NIEAKTYWNA'	Status_oferty	Oferta_sprzedazy
TypKoniaD	VarChar(20)	'GORACOKRWISTY', 'ZIMNOKRWISTY', 'KUC'	Typ_konia	Kon

3.4. Dodatkowe reguły integralnościowe

Dla każdego atrybutu został określony format dopuszczalnych wartości, jakie dany atrybut może przyjmować. Ponadto dla niektórych atrybutów, które mogą przyjmować jedynie wartości z bardzo ograniczonego zbioru, zostały zdefiniowane odpowiednie dziedziny. W ten sposób system będzie zabezpieczony przed próbami wpisania niedozwolonych wartości w wybrane pola.

Każda encja posiada klucz główny będący atrybutem prostym w postaci unikatowego identyfikatora numerycznego. Dzięki temu każda krotka w tabeli będzie unikatowa.

Połączenia miedzy encjami zostały stworzone w taki sposób, by nie występowała redundancja danych. W ten sposób można zapobiec występowaniu anomalii modyfikacji danych, gdyż każda informacja jest przechowywana dokładnie w jednym miejscu.

Stadnina jest główną encją w projekcie i dlatego jest encją obowiązkową w każdym związku, w który wchodzi. Dzięki temu, żeby wprowadzić do bazy nowego pracownika/zwierzę/obiekt konieczne jest przypisanie nowego obiektu do konkretnej stadniny. Jest to zgodne z założeniami projektu, gdyż nie są dopuszczalni pracownicy/zwierzęta/obiekty nieprzynależące do żadnej stadniny. W podobny sposób nie jest możliwe zdefiniowanie nowej oferty, gdy stadnina nie posiada żadnego konia. To również spełnia wymogi systemu i nie powoduje powstania anomalii dołączania. Funkcje te będą zrealizowane przez wymuszenie obowiązkowości klucza obcego.

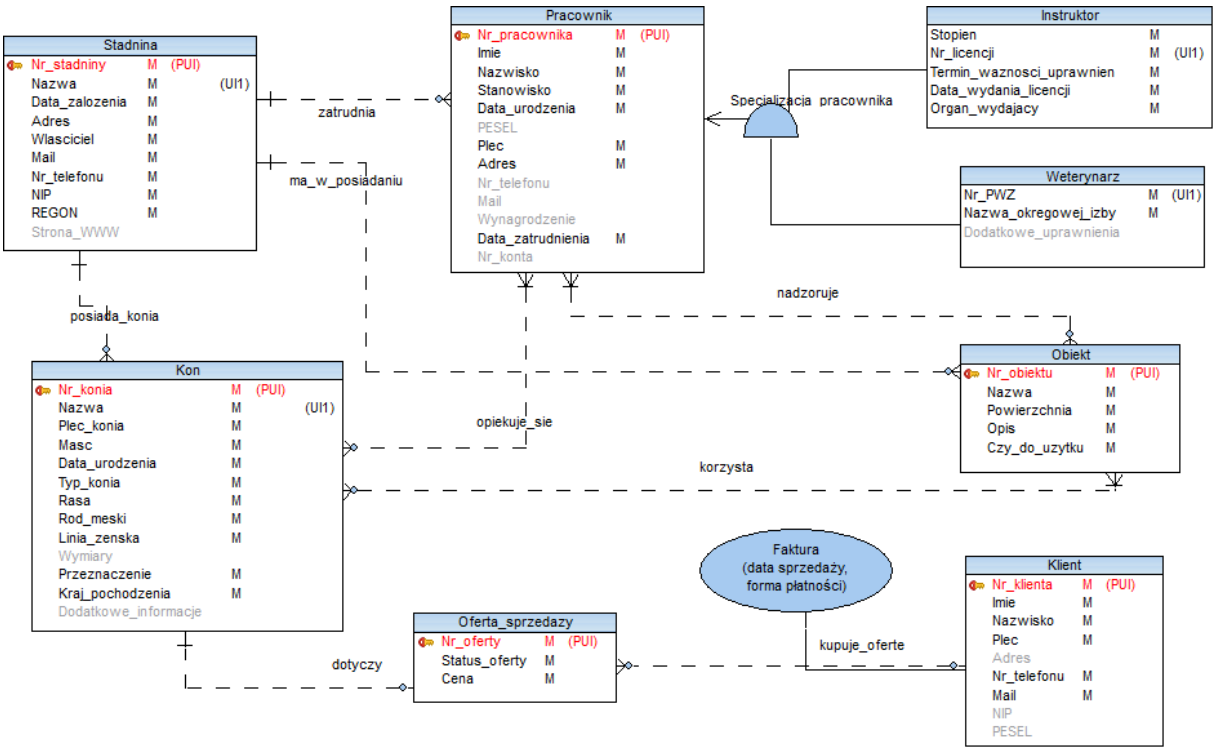
Żeby móc przechowywać w bazie wszystkie wymagane informacje konieczne jest wykorzystanie związków typu n:m. Pozwoli to w fazie logicznej projektu na stworzenie dodatkowych tabel łączących, w których będzie można umieszczać informacje obrazujące m.in. aktualny przydział obowiązków do pracowników.

3.5. Klucze kandydujące i główne

Wszystkie encje posiadają klucz główny w postaci atrybutu prostego, który jest unikatowym identyfikatorem numerycznym danej encji. Ponadto cztery encje posiadają klucze kandydujące. Są to: nazwa stadniny, nazwa konia, numer licencji instruktora, numer prawa wykonywania zawodu (PWZ) weterynarza.

Rodzaj identyfikatora (UI)	Nazwa unikatowego identyfikatora (UI)	Nazwa encji	Nazwa atrybutu
PUI	Stadnina_PK	Stadnina	Nr_stadniny
UI	Stadnina_AK	Stadnina	Nazwa
PUI	Klient_PK	Klient	Nr_klienta
PUI	Kon_PK	Kon	Nr_konia
UI	Kon_AK	Kon	Nazwa
PUI	Obiekt_PK	Obiekt	Nr_obiektu
PUI	Oferta_PK	Oferta_sprzedazy	Nr_oferty
PUI	Pracownik_PK	Pracownik	Nr_pracownika
UI	Instruktor_AK	Instruktor	Nr_licencji
UI	Weterynarz_AK	Weterynarz	Nr_PWZ

3.6. Schemat ER na poziomie konceptualnym



3.7. Problem pułapek szczelinowych i wachlarzowych

Pułapka szczelinowa mogłaby się pojawić między encjami ‘Oferta_sprzedazy’ oraz ‘Klient’, ponieważ minimalne krotności między nimi wynoszą zero. W takiej sytuacji mogłoby dojść np. do utraty informacji, do której stadniny należy dany klient. Jest to jednak zgodne z założeniami projektu, gdyż w bazie mogą być przechowywane informacje o potencjalnym kliencie, który jeszcze nic nie kupił.

Gdyby w modelu zabrakło niektórych połączeń mogłyby powstać pułapki wachlarzowe. Przykładowo, gdyby nie istniał bezpośredni związek miedzy pracownikiem a koniem (*„pracownik opiekuje się koniem”*), a istniałyby jedynie związki *„stadnina posiada konia”* oraz *„stadnina zatrudnia pracownika”,* to nie byłoby możliwe wskazanie, który pracownik jest odpowiedzialny za które konie.

4. Model logiczny

4.1. Charakterystyka modelu relacyjnego

Przejście do modelu relacyjnego wymaga wprowadzenia pewnych zmian w stosunku do modelu konceptualnego.

Encje z poziomu konceptualnego stają się relacjami, dlatego trzeba zmienić ich nazwy z liczby pojedynczej na mnogą.

Związki typu n:m nie są kompatybilne z modelem relacyjnym, dlatego każdy taki związek trzeba zamienić na dwa związki typu 1:n oraz stworzyć nową tabelę łączącą te dwie relacje. Ta tabela przejmie klucze główne z encji, które były w związku n:m i stworzy z nich własny złożony klucz główny.

4.2. Usunięcie niekompatybilności z modelem relacyjnym

Model konceptualny posiadał trzy związki typu n:m. Na etapie projektowania logicznego zostały one zamienione na związki typu 1:n. W wyniku tej operacji powstały nowe relacje.

- ***Relacja 'Konie_Obiekty'***

Atrybuty

Klucz	Nazwa atrybutu	Typ danych	Czy obowiązkowy	Odziedziczony po
PFK	Nr_konia	Integer	T	Konie
PFK	Nr_obiektu	Integer	T	Obiekty

Związki

Nazwa związku	Rodzaj	Relacja (rodzic)	Relacja (dziecko)	Typ i krotność
korzysta	Zidentyfikowany	Konie	Konie_Obiekty	1..1 – 1..N
jest_uzywany	Zidentyfikowany	Obiekty	Konie_Obiekty	1..1 – 0..N

- ***Relacja 'Opieka_nad_konmi'***

Atrybuty

Klucz	Nazwa atrybutu	Typ danych	Czy obowiązkowy	Odziedziczony po
PFK	Nr_pracownika	Integer	T	Pracownicy
PFK	Nr_konia	Integer	T	Konie

Związki

Nazwa związku	Rodzaj	Relacja (rodzic)	Relacja (dziecko)	Typ i krotność
opiekuje_sie	Zidentyfikowany	Pracownicy	Opieka_nad_konmi	1..1 – 0..N
jest_pod_opieka	Zidentyfikowany	Konie	Opieka_nad_konmi	1..1 – 1..N

- ***Relacja 'Pracownicy_Obiekty'***

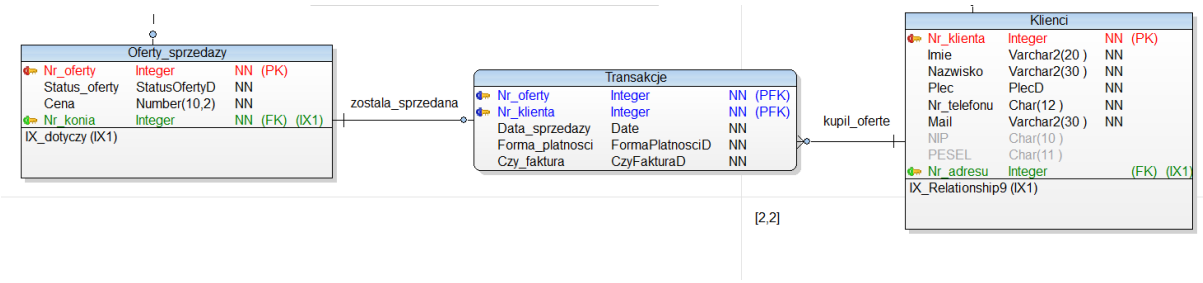
Atrybuty

Klucz	Nazwa atrybutu	Typ danych	Czy obowiązkowy	Odziedziczony po
PFK	Nr_pracownika	Integer	T	Pracownicy
PFK	Nr_obiektu	Integer	T	Obiekty

Związki

Nazwa związku	Rodzaj	Relacja (rodzic)	Relacja (dziecko)	Typ i krotność
nadzoruje	Zidentyfikowany	Pracownicy	Pracownicy_Obiekty	1..1 – 0..N
jest_nadzorowany	Zidentyfikowany	Obiekty	Pracownicy_Obiekty	1..1 – 1..N

Wprawdzie relacje „Oferty_sprzedazy” oraz „Klienci” były połączone związkiem 1:n, który jest implementowalny w modelu relacyjnym, ale nie pozwalało to uwzględnić wymaganych w systemie informacji np. o dacie zawarcia umowy sprzedaży oraz o formie płatności. Rozwiązano to w ten sposób, że stworzono nową relację „***Transakcje***” łączącą tabele „Oferty_sprzedazy” oraz „Klienci”. Wprowadzono również dziedzinę dla pola „Forma_platnosci”: 'GOTOWKA', 'KARTA', 'PRZELEW' oraz dla pola „Czy_faktura”: ‘T’, ‘N’.



4.3. Związki typu 1:1 i problem kluczy obcych

W modelu relacyjnym istnieje 7 związków typu 1:1.

Trzy z nich związane są z adresami: *Stadnina ma adres*, *Pracownik ma adres*, *Klient ma adres*. Ze względu na fakt, że z relacją „Adresy" połączone są trzy różne tabele, to klucz obcy znajduje się zawsze po stronie stadniny/pracownika/obiektu. W przypadku stadniny oraz pracownika klucz ten jest obowiązkowy, czyli relacja adresy jest relacją silną względem stadniny oraz pracownika. Natomiast w przypadku klienta adres nie jest polem obowiązkowym.

Dwa kolejne związki 1:1 są związane z dwoma specjalizacjami. Instruktor/weterynarz są specjalizacjami pracownika, dlatego klucz obcy pojawi się po stronie specjalizacji, bo są to relacje słabe.

Ostatnie dwa związki 1:1 są związane z ofertami sprzedaży. „Oferty sprzedaży" są relacją słabą względem „Koni", dlatego klucz obcy pojawi się po stronie „Oferty sprzedaży". Podobnie „Transakcje" są relacją słabą względem „Ofert sprzedaży", dlatego klucz obcy pojawi się w „Transakcjach" i stanie się tam częścią klucza głównego.

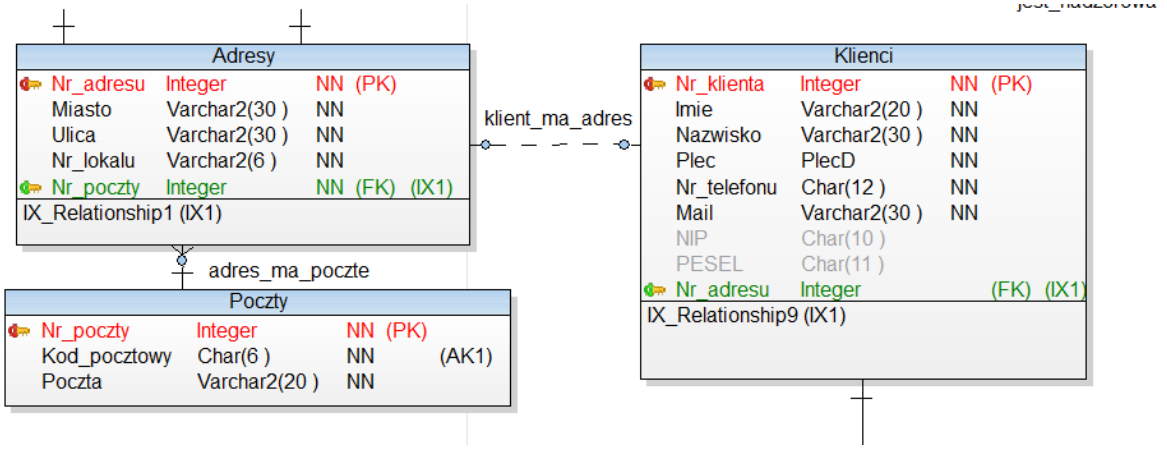
4.4. Proces normalizacji

Pierwsza postać normalna wymaga, by każda wartość atrybutu w każdej krotce była atomowa oraz by nie występowały powtarzające się grupy. W tym celu trzeba pozbyć się pól segmentowych i wielowartościowych oraz powtarzających się grup ze wszystkich relacji.

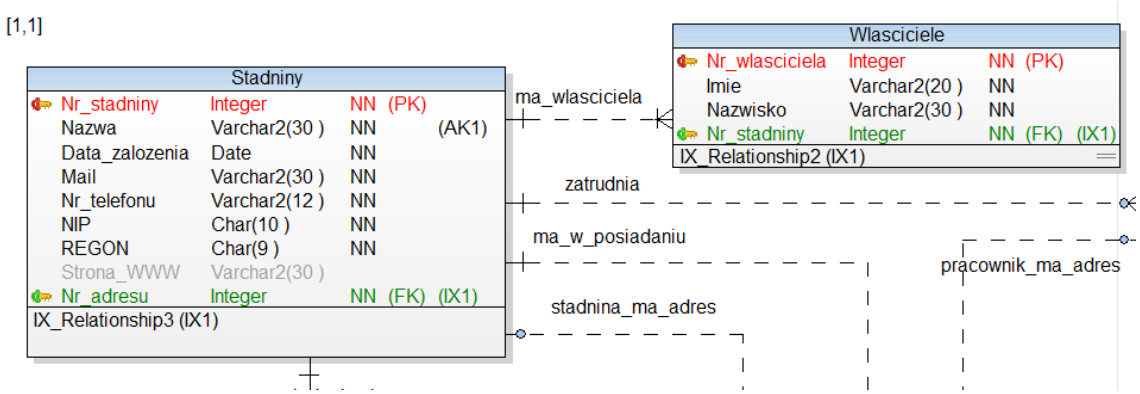
Druga postać normalna wymaga, by wszystkie atrybuty spoza klucza były w pełni funkcyjnie zależne od całego tego klucza. Zapewnia to wykorzystanie kluczy głównych relacji w postaci unikatowych identyfikatorów numerycznych będących atrybutami prostymi.

Trzecia postać normalna będzie spełniona, gdy nie będzie zależności tranzytywnie przechodnich w relacji.

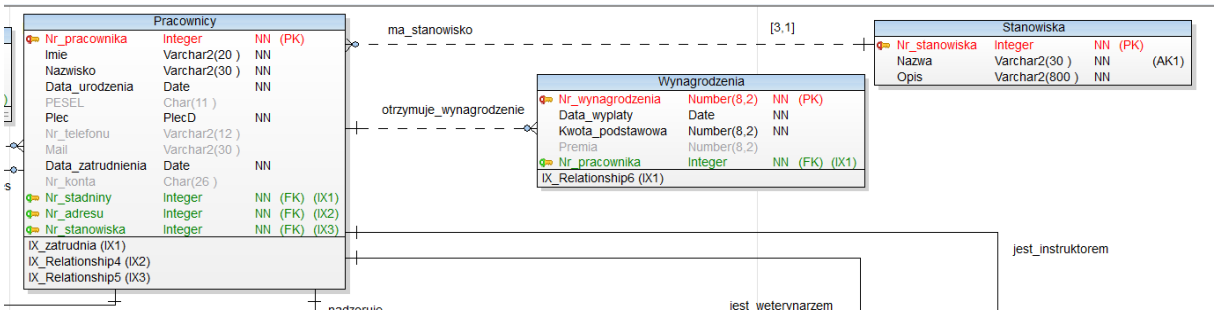
- Adres – pole segmentowe, atrybut pojawia się aż w trzech relacjach: stadniny, klienci, pracownicy. Dlatego zdefiniowane zostały dwa słowniki: dotyczący poczty oraz dotyczący adresu. Pierwszy słownik zawiera unikatowy numer poczty, kod pocztowy oraz nazwę poczty. Drugi zawiera unikatowy identyfikator adresu, nazwę miejscowości, ulicy, numeru lokalu oraz numer poczty (klucz obcy).



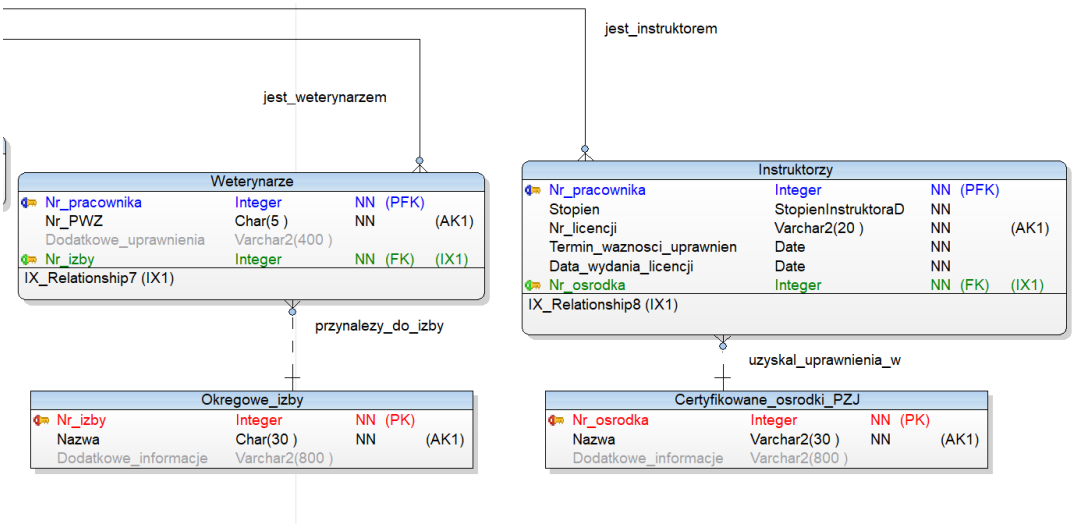
- Właściciele stadniny – pole segmentowe (imię, nazwisko) oraz wielowartościowe (może ich być kilku). Stworzony został nowy słownik zawierający dane o właścicielach. Stadnina jest połączona z nim związkiem 1:n (krotności 1..1 – 1..N).



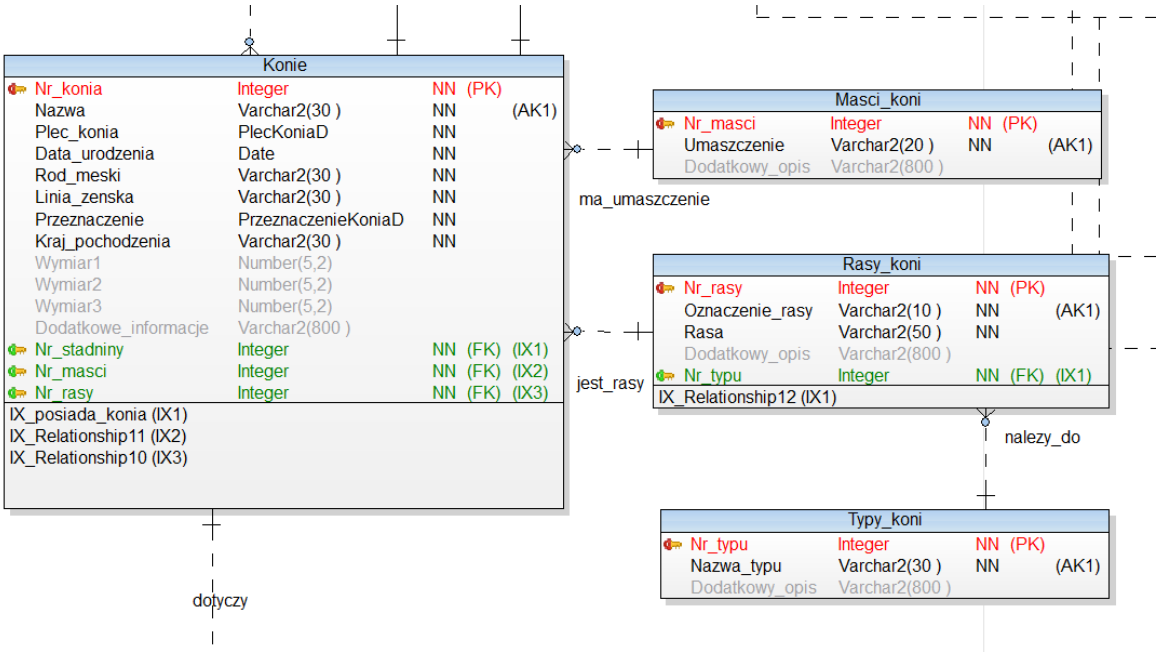
- Stanowiska – może być wielu pracowników na jednym stanowisku i mogą wtedy powstać powtarzające się grupy. Dlatego został stworzony osobny słownik ‘Stanowiska’ zawierający takie pola jak nr_stanowiska (PK), nazwa (AK), opis. Słownik ten jest połączony związkiem 1:n z pracownikiem (krotności 1..1 – 0..N).
- Wynagrodzenia – pole wielowartościowe, gdyż co miesiąc pracownik otrzymuje nowe wynagrodzenie, a system musi przechowywać informacje o historii wypłacanych płac. Dlatego powstała nowa relacja „Wynagrodzenia” zawierająca takie pola jak: nr_wynagrodzenia (PK), data wypłaty, kwota podstawowa, premia, nr_pracownika (FK)



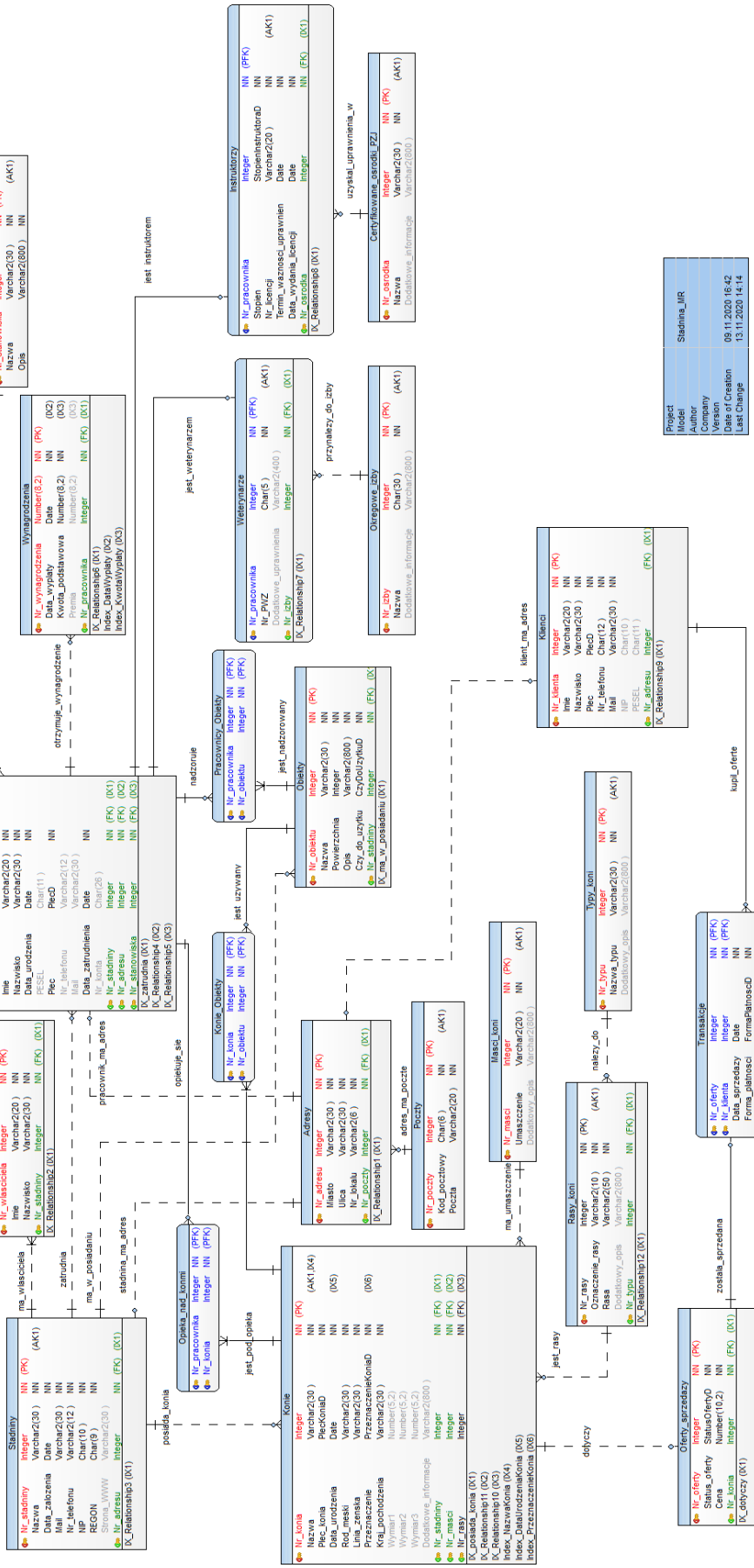
- Stopień instruktora – ograniczony i stały zbiór dopuszczalnych wartości (wyróżnia się pięć kategorii), dlatego zdefiniowano dla tego pola dziedzinę: 'Instruktor Jezdziectwa Powszechnego', 'Instruktor Szkolenia Podstawowego', 'Instruktor Sportu', 'Trener II', 'Trener I'.
- Izby lekarskie oraz ośrodki jeździeckie – kolejne słowniki stworzone po to , by zapobiec powstawaniu powtarzających się grup w relacjach specjalizujących pracownika (weterynarz, instruktor).



- Wymiary konia – pole wielowartościowe (wysokość konia w kłębie, obwód klatki piersiowej, obwód lewej nogi nad pęciną). W celu normalizacji relacji pole to zostało rozbite na trzy pola proste – każde określa jeden parametr zwierzęcia.
- Przeznaczenie konia – ograniczony zbiór wartości, jakie mogą się pojawić, dlatego zdefiniowano dla tego atrybutu nową dziedzinę: 'REKREACYJNE', 'SPORTOWE', 'ROBOCZE', 'HODOWLANE', 'MIESNE', 'ZAPRZEGOWE', 'INNE'.
- Maści konia – atrybut, który ma ograniczony zbiór wartości, ale jednak za duży by definiować dla niego dziedzinę. Dlatego został dla niego dobrany nowy słownik, by nie pojawiały się powtarzające się wartości w relacji „Konie”.
- Rasy i typy konia – kolejne relacje słownikowe. Każdy koń jest określonego typu (rodzina ras) i każdy z takich typów posiada pewien zbiór ras koni.
  W modelu konceptualnym dla atrybutu Typ_konia została określona dziedzina. Zrezygnowano z tego w fazie logicznej, ponieważ znając rasę konia można określić jego typ i tym samym pojawiłaby się zależność tranzytywnie przechodnia w relacji „Konie”. Problem rozwiązano w ten sposób, że stworzono nowe słowniki „Rasy” oraz „Typy” powiązane odpowiednimi związkami między sobą oraz z relacją „Konie”.



4.5. Schemat ER na poziomie logicznym



4.6. Więzy integralności

Integralność encji – każda tabela posiada unikatowy klucz główny będący prostym kluczem numerycznym (Primary Key), jest to pole obowiązkowe każdej relacji (NOT NULL). Dzięki temu jest zapewnione, że w tabeli nie będą mogły istnieć dwa takie same wiersze.

Integralność krotki – dla każdego atrybutu zostały określone dopuszczalne formaty wartości, które można przechowywać w poszczególnych polach oraz rozmiary tych pól. Ponadto dla niektórych atrybutów zawężono dziedzinę, jeśli liczba dopuszczalnych wartości jest stała i niewielka. Większość atrybutów jest obowiązkowa (NOT NULL), niektóre muszą mieć wartości unikatowe (UNIQUE).

Integralność referencyjna (odwołań) – klucz obcy może przyjmować tylko takie wartości klucza głównego, które istnieją w innej tabeli lub wartość NULL, jeżeli klucz obcy jest nieobowiązkowy. W projektowanej bazie istnieje tylko jeden przypadek, gdy klucz obcy jest polem nieobowiązkowym – dotyczy to adresu klienta. Gdy taki adres zostanie usunięty z relacji „Adresy" to w miejsce nr_adresu w tabeli „Klienci" zostanie wstawiona wartość NULL. Zapewni to zaznaczona opcja *SET NULL* ustawiona w polu *Parent DELETE*. W pozostałych przypadkach, gdy zostanie skasowany obiekt nadrzędny (np. koń), to zostanie również skasowany obiekt podrzędny (np. oferta sprzedaży danego konia). Zapewni to zaznaczona opcja *Cascade* w polu *Parent DELETE*. Dzięki temu mechanizmowi zachowana zostanie spójność danych w bazie.

4.7. Proces denormalizacji

W celu zwiększenia wydajności systemu może być opłacalne przeprowadzenie częściowej denormalizacji bazy kosztem wprowadzenia redundancji pewnych danych.

- Redukcja liczby złączeń przez powielenie atrybutów kluczy obcych w związkach 1:n

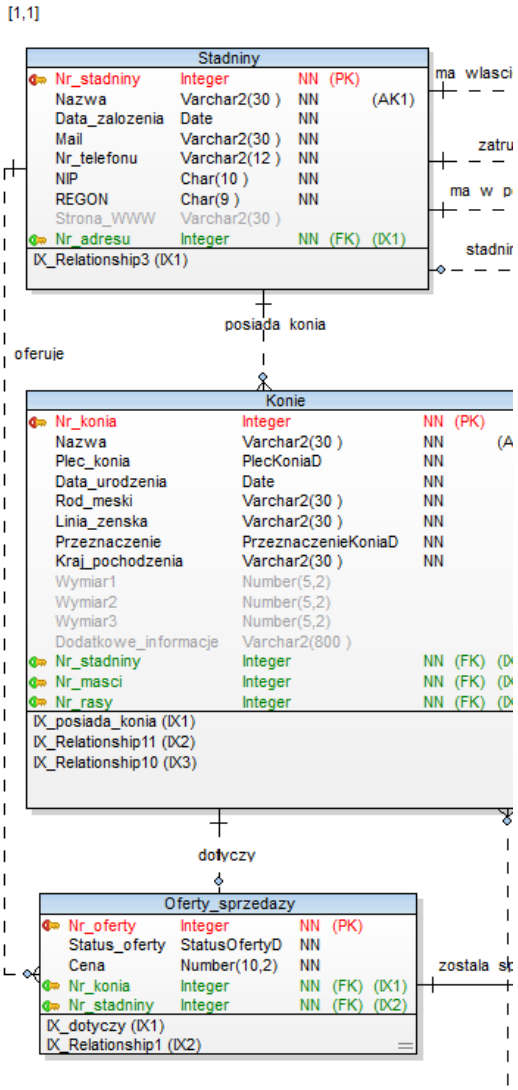
W celu uzyskania informacji, jakie są ceny wszystkich aktywnych ofert sprzedaży, które prowadzi aktualnie stadnina o nazwie ISKRA konieczne jest złączenie aż trzech tabel: „Stadniny", „Konie" oraz „Oferty sprzedaży".

```
SELECT nr_oferty, cena FROM Oferty_sprzedazy NATURAL JOIN Konie NATURAL JOIN Stadniny
WHERE Stadniny.Nazwa='ISKRA' AND Oferty_sprzedazy.Status_oferty='AKTYWNA';
```

Jeśli wprowadzony zostanie dodatkowy związek między stadniną a ofertą: *Stadnina oferuje ofertę sprzedaży*, to liczba złączeń zmniejszy się do dwóch.

```
SELECT nr_oferty, cena FROM Oferty_sprzedazy NATURAL JOIN Stadniny
WHERE Stadniny.Nazwa='ISKRA' AND Oferty_sprzedazy.Status_oferty='AKTYWNA';
```

Jednak powstanie wtedy redundancja danych w bazie, ponieważ klucz obcy *nr_stadniny* będzie nadmiarowy w relacji „Oferty_sprzedazy". Powstanie wtedy cykl, ponieważ będą istniały dwie ścieżki dojścia do tej samej informacji: *Stadniny – Konie – Oferty sprzedaży* oraz *Stadniny – Oferty sprzedaży*. Spowoduje to utratę spójności danych, ale z drugiej strony uprości zapytanie do bazy o poszukiwane informacje.



- Scalenie tabel podglądu z relacjami bazowymi

Gdyby zaistniała potrzeba sprawdzenia, ile koni należących do typu „GORĄCOKRWISTE” i znajdujących się obecnie pod opieką stadniny jest przeznaczonych do celów sportowych, to trzeba by złączyć ze sobą aż trzy tablice: „Konie”, „Rasy koni”, „Typy koni”.

SELECT nr_konia, nazwa FROM Konie NATURAL JOIN Rasy_koni NATURAL JOIN Typy_koni
WHERE Typy_koni.Nazwa_typu=’GORACAKRWISTY’ AND Konie.Przeznaczenie=’SPORTOWE’;

W celu uproszczenia zapytania może być korzystne wprowadzenie tabel słownikowych z powrotem do relacji bazowej. Wówczas wszystkie potrzebne informacje będą zawarte w relacji „Konie”.

SELECT nr_konia, nazwa FROM Konie
WHERE Konie.Nazwa_typu=’GORACAKRWISTY’ AND Konie.Przeznaczenie=’SPORTOWE’;

Z drugiej jednak strony zwiększone zostanie w ten sposób ryzyko, że dane w bazie będą niespójne. Nie będzie wtedy bowiem zabezpieczenia pilnującego, by pracownik wprowadzający do systemu dane o koniu, przypisał zwierzęciu takie parametry jak typ oraz rasa, by były one ze sobą w zgodzie, czyli by wzajemnie się nie wykluczały (np. koń należący do rasy Angloarab nie może mieć jednocześnie typu zimnokrwisty).

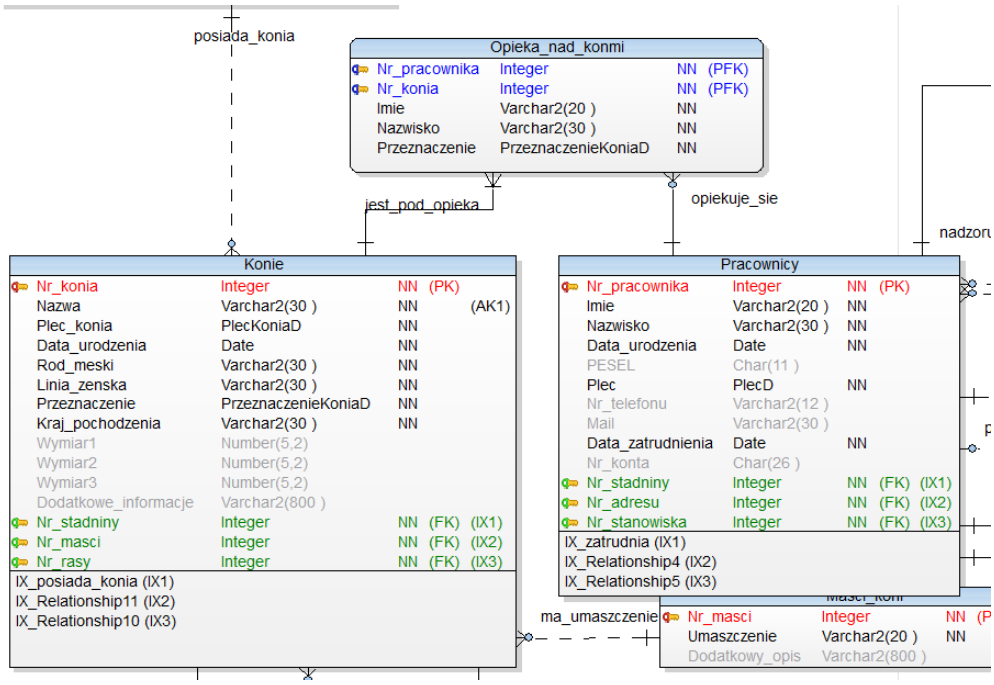
- Redukcja liczby złączeń przez powielenie atrybutów w związkach n:m

Mogłaby zajść potrzeba uzyskania informacji, jak nazywają się wszyscy pracownicy, którzy opiekują się końmi przeznaczonymi do rekreacji. Relacja „Opieka nad końmi" nie zawiera wszystkich tych informacji, dlatego trzeba by przeprowadzić złączenie tabel „Opieka nad końmi", „Pracownicy", „Konie".

SELECT nr_pracownika, imie, nazwisko FROM Opieka_nad_konmi NATURAL JOIN Konie NATURAL JOIN Pracownicy WHERE Konie.Przeznaczenie='REKREACYJNE';

W celu redukcji liczby złączeń można by rozszerzyć atrybuty tabeli łączącej w taki sposób, by oprócz odziedziczonych kluczy głównych zawierała też pewne atrybuty z relacji, które ten związek n:m tworzyły. W rozważanym przypadku byłyby to informacje: „przeznaczenie" z relacji „Konie" oraz „imię", „nazwisko" z relacji „Pracownicy". Dzięki temu najważniejsze informacje, do których najczęściej zachodziłaby potrzeba odwołania się, byłyby umieszczone w jednej tabeli.

SELECT nr_pracownika, imie, nazwisko FROM Opieka_nad_konmi
WHERE Opieka_nad_konmi.Przeznaczenie='REKREACYJNE';



Ale ponownie powstałaby w ten sposób redundancja danych i baza mogłaby utracić spójność.

Projektowana baza nie jest na tyle rozbudowana, by jednoznacznie opłacało się przeprowadzić pewne kroki denormalizacji w celu poprawy wydajności. W perspektywie tego systemu ważniejsze wydaje się zachowanie spójności danych w bazie. Dlatego ostatecznie nie zdecydowano się na żaden z proponowanych kroków.

5. Faza fizyczna

5.1. Projekt transakcji i weryfikacja ich wykonalności

Poziomem izolacji dla transakcji będzie domyślny w bazach Oracle READ COMMITED. Każda transakcja będzie mogła odczytywać tylko te dane, które zostały zatwierdzone, dlatego nie wystąpi anomalia niezatwierdzonego odczytu. Jednak nie zapobiegnie to anomaliom niepowtarzalnego odczytu, ponieważ transakcja po wykonaniu odczytu od razu zwolni blokadę typu S nałożoną na te dane, nie czekając do końca transakcji i tym samym zezwoli innym transakcjom np. na modyfikację tych danych.

W poniższej tabelce przedstawiono główne transakcje, wymagane dla nich tabele oraz rodzaje blokad, jakie trzeba nałożyć na dane zasoby, by przeprowadzić żądaną operację. Tabelka uwzględnia wszystkie relacje, jakie dana transakcja mogłaby potrzebować, jednak w niektórych przypadkach ich liczba może być mniejsza (np. może zajść potrzeba odczytania tylko imienia i nazwiska pewnego pracownika oraz jego aktualnych obowiązków, podczas gdy informacje o jego adresie nie będą obecnie potrzebne).

Transakcja	***Wymagane relacje***	***Rodzaj blokady***
Dodanie/usunięcie pracownika, modyfikacja danych	Pracownicy, Stanowiska, Adresy, Poczty; w przypadku specjalizacji również: Weterynarze, Instruktorzy	X
Wprowadzenie danych o wynagrodzeniu	Wynagrodzenia	X
Przypisanie pracownika do danego zwierzęcia	Opieka_nad_konmi	X
Przypisanie pracownika do danego obiektu	Pracownicy_Obiekty	X
Przeglądanie informacji o pracownikach	Pracownicy, Stanowiska, Wynagrodzenia, Adresy, Poczty, Opieka_nad_konmi, Pracownicy_Obiekty	S
Tworzenie nowych/usuwanie ofert sprzedaży, modyfikowanie oferty	Oferty_sprzedazy	X
Obsługa sprzedanych ofert, przypisanie klienta do danej transakcji	Transakcje	X
Dodanie nowego/usunięcie obiektu z bazy, modyfikacja danych o obiekcie	Obiekty	X
Przeglądanie informacji o obiektach	Obiekty, Pracownicy_Obiekty, Konie_Obiekty	S
Dodanie/usunięcie zwierzęcia z bazy, modyfikacja danych o zwierzęciu	Konie, Masci_koni, Rasy_koni	X
Przypisanie zwierzęcia do danego obiektu	Konie_Obiekty	X
Przeglądanie informacji o zwierzętach	Konie, Masci_koni, Rasy_koni, Konie_Obiekty, Opieka_nad_konmi	S
Dodanie klienta do bazy, modyfikacja danych	Klienci, Adresy, Poczty	X
Przeglądanie informacji o klientach	Klienci, Adresy, Poczty, Transakcje	S
Modyfikacja danych o stadninie	Stadniny, Adresy, Poczty, Wlasciciele	X
Dodanie/usunięcie stadniny	Stadniny, Adresy, Poczty, Wlasciciele	X

W przypadku relacji słownikowych takich jak „Typy_koni”, „Okregowe_izby”, „Certyfikowane_osrodki_PZJ” zakłada się, że będą to dane w zasadzie stałe, zdefiniowane raz przy zakładaniu bazy i jeśli miałaby zajść potrzeba wprowadzenia w nich jakiejś zmiany (np. po założeniu nowego ośrodka) to zdarzałoby się to bardzo rzadko, dlatego nie uwzględniono ich w powyższej tabelce.

Przyjęte założenia pozwolą na poprawne wykonanie wszystkich zamierzonych w systemie transakcji.

5.2. Strojenia bazy danych – dobór indeksów

Oprócz indeksów związanych z obecnością w relacji kluczy obcych stworzono również kilka dodatkowych indeksów, które pozwolą na szybki dostęp do najczęściej poszukiwanych danych w bazie. Jest to szczególnie opłacalne w przypadku dużych, rozbudowanych relacji (takich jak „Konie”), by przyspieszyć wyciąganie z nich istotnych informacji.

Własne indeksy	**Czy UNIQUE**	**Nazwa relacji**	**Nazwy atrybutów**	**Przykładowe zastosowanie**
Index_DataUrodzeniaKonia	N	Konie	Data_urodzenia	Wyszukiwanie koni urodzonych przed danym rokiem
Index_NazwaKonia	T	Konie	Nazwa	Wyszukiwanie informacji o koniu po jego nazwie a nie numerze
Index_PrzeznaczenieKonia	N	Konie	Przeznaczenie	Wyszukiwanie koni według ich przeznaczenia
Index_DataWyplaty	N	Wynagrodzenia	Data_wyplaty	Sumowanie wypłat danego pracownika w perspektywie danego roku
Index_KwotaWyplaty	N	Wynagrodzenia	Kwota_podstawowa, Premia	Sumowanie wypłat danego pracownika w perspektywie danego roku

5.3. Skrypt SQL zakładający bazę danych

```
/*
Created: 09.11.2020
Modified: 13.11.2020
Model: Stadnina_MR
Database: Oracle 19c
*/


-- Create sequences section -------------------------------------------------

CREATE SEQUENCE StadninaSeq1
 INCREMENT BY 1
 START WITH 1
 NOMAXVALUE
 NOMINVALUE
 NOCACHE
/

CREATE SEQUENCE PracownikSeq1
 INCREMENT BY 1
 START WITH 1
 NOMAXVALUE
 NOMINVALUE
 NOCACHE
/

CREATE SEQUENCE AdresSeq1
 INCREMENT BY 1
 START WITH 1
 NOMAXVALUE
 NOMINVALUE
 NOCACHE
/

CREATE SEQUENCE PocztaSeq1
 INCREMENT BY 1
 START WITH 1
 NOMAXVALUE
 NOMINVALUE
 NOCACHE
/

CREATE SEQUENCE WlascicielSeq1
 INCREMENT BY 1
 START WITH 1
 NOMAXVALUE
 NOMINVALUE
 NOCACHE
/
```

```
CREATE SEQUENCE StanowiskoSeq1
 INCREMENT BY 1
 START WITH 1
 NOMAXVALUE
 NOMINVALUE
 NOCACHE
/

CREATE SEQUENCE WynagrodzenieSeq1
 INCREMENT BY 1
 START WITH 1
 NOMAXVALUE
 NOMINVALUE
 NOCACHE
/

CREATE SEQUENCE IzbaSeq1
 INCREMENT BY 1
 START WITH 1
 NOMAXVALUE
 NOMINVALUE
 NOCACHE
/

CREATE SEQUENCE OsrodekSeq1
 INCREMENT BY 1
 START WITH 1
 NOMAXVALUE
 NOMINVALUE
 NOCACHE
/

CREATE SEQUENCE KonSeq1
 INCREMENT BY 1
 START WITH 1
 NOMAXVALUE
 NOMINVALUE
 NOCACHE
/

CREATE SEQUENCE OfertaSeq1
 INCREMENT BY 1
 START WITH 1
 NOMAXVALUE
 NOMINVALUE
 NOCACHE
/

CREATE SEQUENCE RasaSeq1
 INCREMENT BY 1
 START WITH 1
 NOMAXVALUE
```

```
 NOMINVALUE
 NOCACHE
/

CREATE SEQUENCE TypKoniaSeq1
 INCREMENT BY 1
 START WITH 1
 NOMAXVALUE
 NOMINVALUE
 NOCACHE
/

CREATE SEQUENCE MascSeq1
 INCREMENT BY 1
 START WITH 1
 NOMAXVALUE
 NOMINVALUE
 NOCACHE
/

CREATE SEQUENCE KlientSeq1
 INCREMENT BY 1
 START WITH 1
 NOMAXVALUE
 NOMINVALUE
 NOCACHE
/

CREATE SEQUENCE ObiektSeq1
 INCREMENT BY 1
 START WITH 1
 NOMAXVALUE
 NOMINVALUE
 NOCACHE
/

-- Create tables section -------------------------------------------------

-- Table Stadniny

CREATE TABLE Stadniny(
  Nr_stadniny Integer NOT NULL,
  Nazwa Varchar2(30 ) NOT NULL,
  Data_zalozenia Date NOT NULL,
  Mail Varchar2(30 ) NOT NULL,
  Nr_telefonu Varchar2(12 ) NOT NULL,
  NIP Char(10 ) NOT NULL,
  REGON Char(9 ) NOT NULL,
  Strona_WWW Varchar2(30 ),
  Nr_adresu Integer NOT NULL
)
/
```

```
-- Create indexes for table Stadniny

CREATE INDEX IX_Relationship3 ON Stadniny (Nr_adresu)
/

-- Add keys for table Stadniny

ALTER TABLE Stadniny ADD CONSTRAINT Stadnina_PK PRIMARY KEY (Nr_stadniny)
/

ALTER TABLE Stadniny ADD CONSTRAINT Nazwa_stadniny_AK UNIQUE (Nazwa)
/

-- Table Pracownicy

CREATE TABLE Pracownicy(
  Nr_pracownika Integer NOT NULL,
  Imie Varchar2(20 ) NOT NULL,
  Nazwisko Varchar2(30 ) NOT NULL,
  Data_urodzenia Date NOT NULL,
  PESEL Char(11 ),
  Plec Char(1 ) NOT NULL
       CHECK (Plec IN ('K','M')),
  Nr_telefonu Varchar2(12 ),
  Mail Varchar2(30 ),
  Data_zatrudnienia Date NOT NULL,
  Nr_konta Char(26 ),
  Nr_stadniny Integer NOT NULL,
  Nr_adresu Integer NOT NULL,
  Nr_stanowiska Integer NOT NULL
)
/

-- Create indexes for table Pracownicy

CREATE INDEX IX_zatrudnia ON Pracownicy (Nr_stadniny)
/

CREATE INDEX IX_Relationship4 ON Pracownicy (Nr_adresu)
/

CREATE INDEX IX_Relationship5 ON Pracownicy (Nr_stanowiska)
/

-- Add keys for table Pracownicy

ALTER TABLE Pracownicy ADD CONSTRAINT Pracownik_PK PRIMARY KEY (Nr_pracownika)
/

-- Table Instruktorzy
```

```
CREATE TABLE Instruktorzy(
  Nr_pracownika Integer NOT NULL,
  Stopien Varchar2(40 ) NOT NULL
      CHECK (Stopien IN ('Instruktor Jezdziectwa Powszechnego', 'Instruktor Szkolenia
Podstawowego', 'Instruktor Sportu', 'Trener II', 'Trener I')),
  Nr_licencji Varchar2(20 ) NOT NULL,
  Termin_waznosci_uprawnien Date NOT NULL,
  Data_wydania_licencji Date NOT NULL,
  Nr_osrodka Integer NOT NULL
)
/

-- Create indexes for table Instruktorzy

CREATE INDEX IX_Relationship8 ON Instruktorzy (Nr_osrodka)
/

-- Add keys for table Instruktorzy

ALTER TABLE Instruktorzy ADD CONSTRAINT Unique_Identifier1 PRIMARY KEY (Nr_pracownika)
/

ALTER TABLE Instruktorzy ADD CONSTRAINT Nr_licencji UNIQUE (Nr_licencji)
/

-- Table Weterynarze

CREATE TABLE Weterynarze(
  Nr_pracownika Integer NOT NULL,
  Nr_PWZ Char(5 ) NOT NULL,
  Dodatkowe_uprawnienia Varchar2(400 ),
  Nr_izby Integer NOT NULL
)
/

-- Create indexes for table Weterynarze

CREATE INDEX IX_Relationship7 ON Weterynarze (Nr_izby)
/

-- Add keys for table Weterynarze

ALTER TABLE Weterynarze ADD CONSTRAINT Unique_Identifier2 PRIMARY KEY (Nr_pracownika)
/

ALTER TABLE Weterynarze ADD CONSTRAINT Nr_PWZ UNIQUE (Nr_PWZ)
/

-- Table Oferty_sprzedazy

CREATE TABLE Oferty_sprzedazy(
  Nr_oferty Integer NOT NULL,
```

```
  Status_oferty Varchar2(20 ) NOT NULL
      CHECK (Status_oferty IN ('AKTYWNA', 'SPRZEDANA', 'NIEAKTYWNA')),
  Cena Number(10,2) NOT NULL,
  Nr_konia Integer NOT NULL
)
/

-- Create indexes for table Oferty_sprzedazy

CREATE INDEX IX_dotyczy ON Oferty_sprzedazy (Nr_konia)
/

-- Add keys for table Oferty_sprzedazy

ALTER TABLE Oferty_sprzedazy ADD CONSTRAINT Oferta_PK PRIMARY KEY (Nr_oferty)
/

-- Table Konie

CREATE TABLE Konie(
  Nr_konia Integer NOT NULL,
  Nazwa Varchar2(30 ) NOT NULL,
  Plec_konia Varchar2(10 ) NOT NULL
      CHECK (Plec_konia IN ('OGIER', 'KLACZ', 'WALACH')),
  Data_urodzenia Date NOT NULL,
  Rod_meski Varchar2(30 ) NOT NULL,
  Linia_zenska Varchar2(30 ) NOT NULL,
  Przeznaczenie Varchar2(20 ) NOT NULL
      CHECK (Przeznaczenie IN ('REKREACYJNE', 'SPORTOWE', 'ROBOCZE', 'HODOWLANE', 'MIESNE',
'ZAPRZEGOWE', 'INNE')),
  Kraj_pochodzenia Varchar2(30 ) NOT NULL,
  Wymiar1 Number(5,2),
  Wymiar2 Number(5,2),
  Wymiar3 Number(5,2),
  Dodatkowe_informacje Varchar2(800 ),
  Nr_stadniny Integer NOT NULL,
  Nr_masci Integer NOT NULL,
  Nr_rasy Integer NOT NULL
)
/

-- Create indexes for table Konie

CREATE INDEX IX_posiada_konia ON Konie (Nr_stadniny)
/

CREATE INDEX IX_Relationship11 ON Konie (Nr_masci)
/

CREATE INDEX IX_Relationship10 ON Konie (Nr_rasy)
/
```

```
CREATE UNIQUE INDEX Index_NazwaKonia ON Konie (Nazwa)
/

CREATE INDEX Index_DataUrodzeniaKonia ON Konie (Data_urodzenia)
/

CREATE INDEX Index_PrzeznaczenieKonia ON Konie (Przeznaczenie)
/

-- Add keys for table Konie

ALTER TABLE Konie ADD CONSTRAINT Kon_PK PRIMARY KEY (Nr_konia)
/

ALTER TABLE Konie ADD CONSTRAINT Nazwa_konia_AK UNIQUE (Nazwa)
/

-- Table and Columns comments section

COMMENT ON COLUMN Konie.Wymiar1 IS 'Wysokość konia w kłębie w cm'
/
COMMENT ON COLUMN Konie.Wymiar2 IS 'Obwód klatki piersiowej w cm'
/
COMMENT ON COLUMN Konie.Wymiar3 IS 'Obwód lewej nogi nad pęciną w cm'
/

-- Table Klienci

CREATE TABLE Klienci(
  Nr_klienta Integer NOT NULL,
  Imie Varchar2(20 ) NOT NULL,
  Nazwisko Varchar2(30 ) NOT NULL,
  Plec Char(1 ) NOT NULL
      CHECK (Plec IN ('K','M')),
  Nr_telefonu Char(12 ) NOT NULL,
  Mail Varchar2(30 ) NOT NULL,
  NIP Char(10 ),
  PESEL Char(11 ),
  Nr_adresu Integer
)
/

-- Create indexes for table Klienci

CREATE INDEX IX_Relationship9 ON Klienci (Nr_adresu)
/

-- Add keys for table Klienci

ALTER TABLE Klienci ADD CONSTRAINT Klient_PK PRIMARY KEY (Nr_klienta)
/
```

```
-- Table Obiekty

CREATE TABLE Obiekty(
  Nr_obiektu Integer NOT NULL,
  Nazwa Varchar2(30 ) NOT NULL,
  Powierzchnia Integer NOT NULL,
  Opis Varchar2(800 ) NOT NULL,
  Czy_do_uzytku Char(1 ) NOT NULL
      CHECK (Czy_do_uzytku IN ('T','N')),
  Nr_stadniny Integer NOT NULL
)
/

-- Create indexes for table Obiekty

CREATE INDEX IX_ma_w_posiadaniu ON Obiekty (Nr_stadniny)
/

-- Add keys for table Obiekty

ALTER TABLE Obiekty ADD CONSTRAINT Obiekt_PK PRIMARY KEY (Nr_obiektu)
/

-- Table and Columns comments section

COMMENT ON COLUMN Obiekty.Czy_do_uzytku IS 'Czy obiekt nadaje się do użytku'
/

-- Table Opieka_nad_konmi

CREATE TABLE Opieka_nad_konmi(
  Nr_pracownika Integer NOT NULL,
  Nr_konia Integer NOT NULL
)
/

-- Table Pracownicy_Obiekty

CREATE TABLE Pracownicy_Obiekty(
  Nr_pracownika Integer NOT NULL,
  Nr_obiektu Integer NOT NULL
)
/

-- Table Konie_Obiekty

CREATE TABLE Konie_Obiekty(
  Nr_konia Integer NOT NULL,
  Nr_obiektu Integer NOT NULL
)
/
```

```
-- Table Adresy

CREATE TABLE Adresy(
  Nr_adresu Integer NOT NULL,
  Miasto Varchar2(30 ) NOT NULL,
  Ulica Varchar2(30 ) NOT NULL,
  Nr_lokalu Varchar2(6 ) NOT NULL,
  Nr_poczty Integer NOT NULL
)
/

-- Create indexes for table Adresy

CREATE INDEX IX_Relationship1 ON Adresy (Nr_poczty)
/

-- Add keys for table Adresy

ALTER TABLE Adresy ADD CONSTRAINT Adres_PK PRIMARY KEY (Nr_adresu)
/

-- Table and Columns comments section

COMMENT ON COLUMN Adresy.Nr_adresu IS 'Unikatowy numer adresu'
/
COMMENT ON COLUMN Adresy.Miasto IS 'Nazwa miasta'
/
COMMENT ON COLUMN Adresy.Ulica IS 'Nazwa ulicy'
/
COMMENT ON COLUMN Adresy.Nr_lokalu IS 'Numer lokalu'
/

-- Table Poczty

CREATE TABLE Poczty(
  Nr_poczty Integer NOT NULL,
  Kod_pocztowy Char(6 ) NOT NULL,
  Poczta Varchar2(20 ) NOT NULL
)
/

-- Add keys for table Poczty

ALTER TABLE Poczty ADD CONSTRAINT Poczta_PK PRIMARY KEY (Nr_poczty)
/

ALTER TABLE Poczty ADD CONSTRAINT Kod_pocztowy UNIQUE (Kod_pocztowy)
/

-- Table and Columns comments section

COMMENT ON COLUMN Poczty.Nr_poczty IS 'Unikatowy numer poczty'
```

```
/
COMMENT ON COLUMN Poczty.Kod_pocztowy IS 'Kod pocztowy'
/
COMMENT ON COLUMN Poczty.Poczta IS 'Nazwa poczty'
/

-- Table Wlasciciele

CREATE TABLE Wlasciciele(
  Nr_wlasciciela Integer NOT NULL,
  Imie Varchar2(20 ) NOT NULL,
  Nazwisko Varchar2(30 ) NOT NULL,
  Nr_stadniny Integer NOT NULL
)
/

-- Create indexes for table Wlasciciele

CREATE INDEX IX_Relationship2 ON Wlasciciele (Nr_stadniny)
/

-- Add keys for table Wlasciciele

ALTER TABLE Wlasciciele ADD CONSTRAINT Wlasciciel_PK PRIMARY KEY (Nr_wlasciciela)
/

-- Table and Columns comments section

COMMENT ON COLUMN Wlasciciele.Nr_wlasciciela IS 'Unikatowy numer właściciela'
/
COMMENT ON COLUMN Wlasciciele.Imie IS 'Imię właściciela'
/
COMMENT ON COLUMN Wlasciciele.Nazwisko IS 'Nazwisko właściciela'
/

-- Table Stanowiska

CREATE TABLE Stanowiska(
  Nr_stanowiska Integer NOT NULL,
  Nazwa Varchar2(30 ) NOT NULL,
  Opis Varchar2(800 ) NOT NULL
)
/

-- Add keys for table Stanowiska

ALTER TABLE Stanowiska ADD CONSTRAINT Stanowisko_PK PRIMARY KEY (Nr_stanowiska)
/

ALTER TABLE Stanowiska ADD CONSTRAINT Nazwa_stanowiska UNIQUE (Nazwa)
/
```

```
-- Table and Columns comments section

COMMENT ON COLUMN Stanowiska.Nr_stanowiska IS 'Unikatowy numer stanowiska'
/
COMMENT ON COLUMN Stanowiska.Nazwa IS 'Nazwa stanowiska'
/
COMMENT ON COLUMN Stanowiska.Opis IS 'Opis stanowiska'
/

-- Table Wynagrodzenia

CREATE TABLE Wynagrodzenia(
  Nr_wynagrodzenia Number(8,2) NOT NULL,
  Data_wyplaty Date NOT NULL,
  Kwota_podstawowa Number(8,2) NOT NULL,
  Premia Number(8,2),
  Nr_pracownika Integer NOT NULL
)
/

-- Create indexes for table Wynagrodzenia

CREATE INDEX IX_Relationship6 ON Wynagrodzenia (Nr_pracownika)
/

CREATE INDEX Index_DataWyplaty ON Wynagrodzenia (Data_wyplaty)
/

CREATE INDEX Index_KwotaWyplaty ON Wynagrodzenia (Kwota_podstawowa,Premia)
/

-- Add keys for table Wynagrodzenia

ALTER TABLE Wynagrodzenia ADD CONSTRAINT Wynagrodzenie_PK PRIMARY KEY
(Nr_wynagrodzenia)
/

-- Table and Columns comments section

COMMENT ON COLUMN Wynagrodzenia.Nr_wynagrodzenia IS 'Unikatowy numer wynagrodzenia'
/
COMMENT ON COLUMN Wynagrodzenia.Data_wyplaty IS 'Data wypłaty'
/
COMMENT ON COLUMN Wynagrodzenia.Kwota_podstawowa IS 'Kwota podstawowa
wynagrodzenia'
/
COMMENT ON COLUMN Wynagrodzenia.Premia IS 'Kwota dodatkowa wynagrodzenia'
/

-- Table Okregowe_izby

CREATE TABLE Okregowe_izby(
```

```
  Nr_izby Integer NOT NULL,
  Nazwa Char(30 ) NOT NULL,
  Dodatkowe_informacje Varchar2(800 )
)
/

-- Add keys for table Okregowe_izby

ALTER TABLE Okregowe_izby ADD CONSTRAINT Izba_PK PRIMARY KEY (Nr_izby)
/

ALTER TABLE Okregowe_izby ADD CONSTRAINT Nazwa_izby UNIQUE (Nazwa)
/

-- Table and Columns comments section

COMMENT ON COLUMN Okregowe_izby.Nr_izby IS 'Unikatowy numer izby lekarsko-
weterynaryjnej'
/
COMMENT ON COLUMN Okregowe_izby.Nazwa IS 'Nazwa izby'
/
COMMENT ON COLUMN Okregowe_izby.Dodatkowe_informacje IS 'Dotatkowy opis izby'
/

-- Table Certyfikowane_osrodki_PZJ

CREATE TABLE Certyfikowane_osrodki_PZJ(
  Nr_osrodka Integer NOT NULL,
  Nazwa Varchar2(30 ) NOT NULL,
  Dodatkowe_informacje Varchar2(800 )
)
/

-- Add keys for table Certyfikowane_osrodki_PZJ

ALTER TABLE Certyfikowane_osrodki_PZJ ADD CONSTRAINT Osrodek_PK PRIMARY KEY
(Nr_osrodka)
/

ALTER TABLE Certyfikowane_osrodki_PZJ ADD CONSTRAINT Nazwa_osrodka UNIQUE (Nazwa)
/

-- Table and Columns comments section

COMMENT ON COLUMN Certyfikowane_osrodki_PZJ.Nr_osrodka IS 'Unikatowy numer ośrodka'
/
COMMENT ON COLUMN Certyfikowane_osrodki_PZJ.Nazwa IS 'Nazwa ośrodka'
/
COMMENT ON COLUMN Certyfikowane_osrodki_PZJ.Dodatkowe_informacje IS 'Dodatkowe
informacje o ośrodku'
/
```

```
-- Table Transakcje

CREATE TABLE Transakcje(
  Nr_oferty Integer NOT NULL,
  Nr_klienta Integer NOT NULL,
  Data_sprzedazy Date NOT NULL,
  Forma_platnosci Varchar2(10 ) NOT NULL
      CHECK (Forma_platnosci IN ('GOTOWKA', 'KARTA', 'PRZELEW')),
  Czy_faktura Char(1 ) NOT NULL
      CHECK (Czy_faktura IN ('T', 'N'))
)
/

-- Add keys for table Transakcje

ALTER TABLE Transakcje ADD CONSTRAINT PK_Transakcje PRIMARY KEY (Nr_oferty,Nr_klienta)
/

-- Table and Columns comments section

COMMENT ON COLUMN Transakcje.Data_sprzedazy IS 'Data sprzedaży'
/
COMMENT ON COLUMN Transakcje.Forma_platnosci IS 'Forma płatności'
/
COMMENT ON COLUMN Transakcje.Czy_faktura IS 'Czy wystawiana faktura'
/

-- Table Masci_koni

CREATE TABLE Masci_koni(
  Nr_masci Integer NOT NULL,
  Umaszczenie Varchar2(20 ) NOT NULL,
  Dodatkowy_opis Varchar2(800 )
)
/

-- Add keys for table Masci_koni

ALTER TABLE Masci_koni ADD CONSTRAINT PK_Masci_koni PRIMARY KEY (Nr_masci)
/

ALTER TABLE Masci_koni ADD CONSTRAINT Nazwa UNIQUE (Umaszczenie)
/

-- Table and Columns comments section

COMMENT ON COLUMN Masci_koni.Nr_masci IS 'Unikatowy numer koloru'
/
COMMENT ON COLUMN Masci_koni.Umaszczenie IS 'Nazwa koloru'
/
COMMENT ON COLUMN Masci_koni.Dodatkowy_opis IS 'Dodatkowe informacje'
/
```

```
-- Table Rasy_koni

CREATE TABLE Rasy_koni(
  Nr_rasy Integer NOT NULL,
  Oznaczenie_rasy Varchar2(10 ) NOT NULL,
  Rasa Varchar2(50 ) NOT NULL,
  Dodatkowy_opis Varchar2(800 ),
  Nr_typu Integer NOT NULL
)
/

-- Create indexes for table Rasy_koni

CREATE INDEX IX_Relationship12 ON Rasy_koni (Nr_typu)
/

-- Add keys for table Rasy_koni

ALTER TABLE Rasy_koni ADD CONSTRAINT PK_Rasy_koni PRIMARY KEY (Nr_rasy)
/

ALTER TABLE Rasy_koni ADD CONSTRAINT Oznaczenie_rasy UNIQUE (Oznaczenie_rasy)
/

-- Table and Columns comments section

COMMENT ON COLUMN Rasy_koni.Nr_rasy IS 'Unikatowy numer rasy'
/
COMMENT ON COLUMN Rasy_koni.Oznaczenie_rasy IS 'Kod oznaczenia rasy'
/
COMMENT ON COLUMN Rasy_koni.Rasa IS 'Nazwa rasy'
/
COMMENT ON COLUMN Rasy_koni.Dodatkowy_opis IS 'Dodatkowy opis'
/

-- Table Typy_koni

CREATE TABLE Typy_koni(
  Nr_typu Integer NOT NULL,
  Nazwa_typu Varchar2(30 ) NOT NULL,
  Dodatkowy_opis Varchar2(800 )
)
/

-- Add keys for table Typy_koni

ALTER TABLE Typy_koni ADD CONSTRAINT PK_Typy_koni PRIMARY KEY (Nr_typu)
/

ALTER TABLE Typy_koni ADD CONSTRAINT Nazwa_typu UNIQUE (Nazwa_typu)
/
```

```
-- Table and Columns comments section

COMMENT ON COLUMN Typy_koni.Nr_typu IS 'Unikatowy numer typu'
/
COMMENT ON COLUMN Typy_koni.Nazwa_typu IS 'Nazwa typu konia'
/
COMMENT ON COLUMN Typy_koni.Dodatkowy_opis IS 'Dodatkowy opis'
/

-- Trigger for sequence StadninaSeq1 for column Nr_stadniny in table Stadniny ---------
CREATE OR REPLACE TRIGGER ts_Stadniny_StadninaSeq1 BEFORE INSERT
ON Stadniny FOR EACH ROW
BEGIN
  :new.Nr_stadniny := StadninaSeq1.nextval;
END;
/
CREATE OR REPLACE TRIGGER tsu_Stadniny_StadninaSeq1 AFTER UPDATE OF Nr_stadniny
ON Stadniny FOR EACH ROW
BEGIN
  RAISE_APPLICATION_ERROR(-20010,'Cannot update column Nr_stadniny in table Stadniny as it
uses sequence.');
END;
/

-- Trigger for sequence PracownikSeq1 for column Nr_pracownika in table Pracownicy ---------
CREATE OR REPLACE TRIGGER ts_Pracownicy_PracownikSeq1 BEFORE INSERT
ON Pracownicy FOR EACH ROW
BEGIN
  :new.Nr_pracownika := PracownikSeq1.nextval;
END;
/
CREATE OR REPLACE TRIGGER tsu_Pracownicy_PracownikSeq1 AFTER UPDATE OF Nr_pracownika
ON Pracownicy FOR EACH ROW
BEGIN
  RAISE_APPLICATION_ERROR(-20010,'Cannot update column Nr_pracownika in table Pracownicy
as it uses sequence.');
END;
/

-- Trigger for sequence OfertaSeq1 for column Nr_oferty in table Oferty_sprzedazy ---------
CREATE OR REPLACE TRIGGER ts_Oferty_sprzedazy_OfertaSeq1 BEFORE INSERT
ON Oferty_sprzedazy FOR EACH ROW
BEGIN
  :new.Nr_oferty := OfertaSeq1.nextval;
END;
/
CREATE OR REPLACE TRIGGER tsu_Oferty_sprzedazy_OfertaSeq1 AFTER UPDATE OF Nr_oferty
ON Oferty_sprzedazy FOR EACH ROW
BEGIN
  RAISE_APPLICATION_ERROR(-20010,'Cannot update column Nr_oferty in table Oferty_sprzedazy
as it uses sequence.');
```

```
END;
/

-- Trigger for sequence KonSeq1 for column Nr_konia in table Konie ---------
CREATE OR REPLACE TRIGGER ts_Konie_KonSeq1 BEFORE INSERT
ON Konie FOR EACH ROW
BEGIN
  :new.Nr_konia := KonSeq1.nextval;
END;
/
CREATE OR REPLACE TRIGGER tsu_Konie_KonSeq1 AFTER UPDATE OF Nr_konia
ON Konie FOR EACH ROW
BEGIN
  RAISE_APPLICATION_ERROR(-20010,'Cannot update column Nr_konia in table Konie as it uses
sequence.');
END;
/

-- Trigger for sequence KlientSeq1 for column Nr_klienta in table Klienci ---------
CREATE OR REPLACE TRIGGER ts_Klienci_KlientSeq1 BEFORE INSERT
ON Klienci FOR EACH ROW
BEGIN
  :new.Nr_klienta := KlientSeq1.nextval;
END;
/
CREATE OR REPLACE TRIGGER tsu_Klienci_KlientSeq1 AFTER UPDATE OF Nr_klienta
ON Klienci FOR EACH ROW
BEGIN
  RAISE_APPLICATION_ERROR(-20010,'Cannot update column Nr_klienta in table Klienci as it uses
sequence.');
END;
/

-- Trigger for sequence ObiektSeq1 for column Nr_obiektu in table Obiekty ---------
CREATE OR REPLACE TRIGGER ts_Obiekty_ObiektSeq1 BEFORE INSERT
ON Obiekty FOR EACH ROW
BEGIN
  :new.Nr_obiektu := ObiektSeq1.nextval;
END;
/
CREATE OR REPLACE TRIGGER tsu_Obiekty_ObiektSeq1 AFTER UPDATE OF Nr_obiektu
ON Obiekty FOR EACH ROW
BEGIN
  RAISE_APPLICATION_ERROR(-20010,'Cannot update column Nr_obiektu in table Obiekty as it
uses sequence.');
END;
/

-- Trigger for sequence AdresSeq1 for column Nr_adresu in table Adresy ---------
CREATE OR REPLACE TRIGGER ts_Adresy_AdresSeq1 BEFORE INSERT
ON Adresy FOR EACH ROW
BEGIN
```

```
  :new.Nr_adresu := AdresSeq1.nextval;
END;
/
CREATE OR REPLACE TRIGGER tsu_Adresy_AdresSeq1 AFTER UPDATE OF Nr_adresu
ON Adresy FOR EACH ROW
BEGIN
  RAISE_APPLICATION_ERROR(-20010,'Cannot update column Nr_adresu in table Adresy as it uses
sequence.');
END;
/

-- Trigger for sequence PocztaSeq1 for column Nr_poczty in table Poczty ---------
CREATE OR REPLACE TRIGGER ts_Poczty_PocztaSeq1 BEFORE INSERT
ON Poczty FOR EACH ROW
BEGIN
  :new.Nr_poczty := PocztaSeq1.nextval;
END;
/
CREATE OR REPLACE TRIGGER tsu_Poczty_PocztaSeq1 AFTER UPDATE OF Nr_poczty
ON Poczty FOR EACH ROW
BEGIN
  RAISE_APPLICATION_ERROR(-20010,'Cannot update column Nr_poczty in table Poczty as it uses
sequence.');
END;
/

-- Trigger for sequence WlascicielSeq1 for column Nr_wlasciciela in table Wlasciciele ---------
CREATE OR REPLACE TRIGGER ts_Wlasciciele_WlascicielSeq1 BEFORE INSERT
ON Wlasciciele FOR EACH ROW
BEGIN
  :new.Nr_wlasciciela := WlascicielSeq1.nextval;
END;
/
CREATE OR REPLACE TRIGGER tsu_Wlasciciele_WlascicielSeq1 AFTER UPDATE OF Nr_wlasciciela
ON Wlasciciele FOR EACH ROW
BEGIN
  RAISE_APPLICATION_ERROR(-20010,'Cannot update column Nr_wlasciciela in table Wlasciciele as
it uses sequence.');
END;
/

-- Trigger for sequence StanowiskoSeq1 for column Nr_stanowiska in table Stanowiska ---------
CREATE OR REPLACE TRIGGER ts_Stanowiska_StanowiskoSeq1 BEFORE INSERT
ON Stanowiska FOR EACH ROW
BEGIN
  :new.Nr_stanowiska := StanowiskoSeq1.nextval;
END;
/
CREATE OR REPLACE TRIGGER tsu_Stanowiska_StanowiskoSeq1 AFTER UPDATE OF Nr_stanowiska
ON Stanowiska FOR EACH ROW
BEGIN
```

```
  RAISE_APPLICATION_ERROR(-20010,'Cannot update column Nr_stanowiska in table Stanowiska
as it uses sequence.');
END;
/

-- Trigger for sequence WynagrodzenieSeq1 for column Nr_wynagrodzenia in table Wynagrodzenia
---------
CREATE OR REPLACE TRIGGER ts_Wynagrodzenia_WynagrodzenieSeq1 BEFORE INSERT
ON Wynagrodzenia FOR EACH ROW
BEGIN
  :new.Nr_wynagrodzenia := WynagrodzenieSeq1.nextval;
END;
/
CREATE OR REPLACE TRIGGER tsu_Wynagrodzenia_WynagrodzenieSeq1 AFTER UPDATE OF
Nr_wynagrodzenia
ON Wynagrodzenia FOR EACH ROW
BEGIN
  RAISE_APPLICATION_ERROR(-20010,'Cannot update column Nr_wynagrodzenia in table
Wynagrodzenia as it uses sequence.');
END;
/

-- Trigger for sequence IzbaSeq1 for column Nr_izby in table Okregowe_izby ---------
CREATE OR REPLACE TRIGGER ts_Okregowe_izby_IzbaSeq1 BEFORE INSERT
ON Okregowe_izby FOR EACH ROW
BEGIN
  :new.Nr_izby := IzbaSeq1.nextval;
END;
/
CREATE OR REPLACE TRIGGER tsu_Okregowe_izby_IzbaSeq1 AFTER UPDATE OF Nr_izby
ON Okregowe_izby FOR EACH ROW
BEGIN
  RAISE_APPLICATION_ERROR(-20010,'Cannot update column Nr_izby in table Okregowe_izby as it
uses sequence.');
END;
/

-- Trigger for sequence OsrodekSeq1 for column Nr_osrodka in table Certyfikowane_osrodki_PZJ --
-------
CREATE OR REPLACE TRIGGER ts_Certyfikowane_osrodki_PZJ_OsrodekSeq1 BEFORE INSERT
ON Certyfikowane_osrodki_PZJ FOR EACH ROW
BEGIN
  :new.Nr_osrodka := OsrodekSeq1.nextval;
END;
/
CREATE OR REPLACE TRIGGER tsu_Certyfikowane_osrodki_PZJ_OsrodekSeq1 AFTER UPDATE OF
Nr_osrodka
ON Certyfikowane_osrodki_PZJ FOR EACH ROW
BEGIN
  RAISE_APPLICATION_ERROR(-20010,'Cannot update column Nr_osrodka in table
Certyfikowane_osrodki_PZJ as it uses sequence.');
END;
```

```
/

-- Trigger for sequence MascSeq1 for column Nr_masci in table Masci_koni ---------
CREATE OR REPLACE TRIGGER ts_Masci_koni_MascSeq1 BEFORE INSERT
ON Masci_koni FOR EACH ROW
BEGIN
  :new.Nr_masci := MascSeq1.nextval;
END;
/
CREATE OR REPLACE TRIGGER tsu_Masci_koni_MascSeq1 AFTER UPDATE OF Nr_masci
ON Masci_koni FOR EACH ROW
BEGIN
  RAISE_APPLICATION_ERROR(-20010,'Cannot update column Nr_masci in table Masci_koni as it
uses sequence.');
END;
/

-- Trigger for sequence RasaSeq1 for column Nr_rasy in table Rasy_koni ---------
CREATE OR REPLACE TRIGGER ts_Rasy_koni_RasaSeq1 BEFORE INSERT
ON Rasy_koni FOR EACH ROW
BEGIN
  :new.Nr_rasy := RasaSeq1.nextval;
END;
/
CREATE OR REPLACE TRIGGER tsu_Rasy_koni_RasaSeq1 AFTER UPDATE OF Nr_rasy
ON Rasy_koni FOR EACH ROW
BEGIN
  RAISE_APPLICATION_ERROR(-20010,'Cannot update column Nr_rasy in table Rasy_koni as it uses
sequence.');
END;
/

-- Trigger for sequence TypKoniaSeq1 for column Nr_typu in table Typy_koni ---------
CREATE OR REPLACE TRIGGER ts_Typy_koni_TypKoniaSeq1 BEFORE INSERT
ON Typy_koni FOR EACH ROW
BEGIN
  :new.Nr_typu := TypKoniaSeq1.nextval;
END;
/
CREATE OR REPLACE TRIGGER tsu_Typy_koni_TypKoniaSeq1 AFTER UPDATE OF Nr_typu
ON Typy_koni FOR EACH ROW
BEGIN
  RAISE_APPLICATION_ERROR(-20010,'Cannot update column Nr_typu in table Typy_koni as it uses
sequence.');
END;
/



-- Create foreign keys (relationships) section ------------------------------------------------

ALTER TABLE Pracownicy ADD CONSTRAINT zatrudnia FOREIGN KEY (Nr_stadniny) REFERENCES
Stadniny (Nr_stadniny) ON DELETE CASCADE
```

```
/


ALTER TABLE Konie ADD CONSTRAINT posiada_konia FOREIGN KEY (Nr_stadniny) REFERENCES
Stadniny (Nr_stadniny) ON DELETE CASCADE
/


ALTER TABLE Obiekty ADD CONSTRAINT ma_w_posiadaniu FOREIGN KEY (Nr_stadniny)
REFERENCES Stadniny (Nr_stadniny) ON DELETE CASCADE
/


ALTER TABLE Oferty_sprzedazy ADD CONSTRAINT dotyczy FOREIGN KEY (Nr_konia) REFERENCES
Konie (Nr_konia) ON DELETE CASCADE
/


ALTER TABLE Adresy ADD CONSTRAINT adres_ma_poczte FOREIGN KEY (Nr_poczty) REFERENCES
Poczty (Nr_poczty) ON DELETE CASCADE
/


ALTER TABLE Wlasciciele ADD CONSTRAINT ma_wlasciciela FOREIGN KEY (Nr_stadniny)
REFERENCES Stadniny (Nr_stadniny) ON DELETE CASCADE
/


ALTER TABLE Stadniny ADD CONSTRAINT stadnina_ma_adres FOREIGN KEY (Nr_adresu)
REFERENCES Adresy (Nr_adresu) ON DELETE CASCADE
/


ALTER TABLE Pracownicy ADD CONSTRAINT pracownik_ma_adres FOREIGN KEY (Nr_adresu)
REFERENCES Adresy (Nr_adresu) ON DELETE CASCADE
/


ALTER TABLE Pracownicy ADD CONSTRAINT ma_stanowisko FOREIGN KEY (Nr_stanowiska)
REFERENCES Stanowiska (Nr_stanowiska) ON DELETE CASCADE
/
```

```
ALTER TABLE Wynagrodzenia ADD CONSTRAINT otrzymuje_wynagrodzenie FOREIGN KEY
(Nr_pracownika) REFERENCES Pracownicy (Nr_pracownika) ON DELETE CASCADE
/



ALTER TABLE Weterynarze ADD CONSTRAINT przynalezy_do_izby FOREIGN KEY (Nr_izby)
REFERENCES Okregowe_izby (Nr_izby) ON DELETE CASCADE
/



ALTER TABLE Instruktorzy ADD CONSTRAINT uzyskal_uprawnienia_w FOREIGN KEY (Nr_osrodka)
REFERENCES Certyfikowane_osrodki_PZJ (Nr_osrodka) ON DELETE CASCADE
/



ALTER TABLE Klienci ADD CONSTRAINT klient_ma_adres FOREIGN KEY (Nr_adresu) REFERENCES
Adresy (Nr_adresu) ON DELETE SET NULL
/



ALTER TABLE Transakcje ADD CONSTRAINT zostala_sprzedana FOREIGN KEY (Nr_oferty)
REFERENCES Oferty_sprzedazy (Nr_oferty) ON DELETE CASCADE
/



ALTER TABLE Transakcje ADD CONSTRAINT kupil_oferte FOREIGN KEY (Nr_klienta) REFERENCES
Klienci (Nr_klienta) ON DELETE CASCADE
/



ALTER TABLE Konie ADD CONSTRAINT ma_umaszczenie FOREIGN KEY (Nr_masci) REFERENCES
Masci_koni (Nr_masci) ON DELETE CASCADE
/



ALTER TABLE Rasy_koni ADD CONSTRAINT nalezy_do FOREIGN KEY (Nr_typu) REFERENCES
Typy_koni (Nr_typu) ON DELETE CASCADE
/



ALTER TABLE Konie ADD CONSTRAINT jest_rasy FOREIGN KEY (Nr_rasy) REFERENCES Rasy_koni
(Nr_rasy) ON DELETE CASCADE
/
```

5.4. Przykłady zapytań i poleceń SQL odnoszących się do bazy

Wprowadzanie danych do bazy należy zacząć od relacji brzegowych typu „Poczty”, „Adresy”, „Stanowiska”, „Maści koni”, „Typy koni”, „Rasy koni”. Następnie można zacząć wprowadzać dane do głównych relacji projektu takich jak „Stadniny”, „Pracownicy”, „Konie”. Poniżej zamieszczono przykłady poleceń, które zostały użyte do wypełnienie bazy.

- Założenie nowej stadniny

INSERT INTO Stadniny (NAZWA, DATA_ZALOZENIA, MAIL, NR_TELEFONU, NIP, REGON, STRONA_WWW, NR_ADRESU) VALUES ('Stadnina Iskra', ‘10/11/10’, 'kontakt@stadninaiskra.pl', '333456987', '5791891126', '221072310', 'stadninaiskra.pl', '2');

- Dodanie kolorów do słownika

INSERT INTO Masci_koni (UMASZCZENIE) VALUES ('kasztanowe');

INSERT INTO Masci_koni (UMASZCZENIE) VALUES ('siwe');

INSERT INTO Masci_koni (UMASZCZENIE) VALUES ('gniade');

- Dodanie nowego pracownika do bazy

INSERT INTO Pracownicy (IMIE, NAZWISKO, DATA_URODZENIA, PESEL, PLEC, NR_TELEFONU, MAIL, DATA_ZATRUDNIENIA, NR_STADNINY, NR_ADRESU, NR_STANOWISKA) VALUES ('Mariusz', 'Wiosenny', ‘78/12/12’, '78121209874', 'M', '223567910', 'm.wiosenny@stajniaiskra.pl', ‘90/04/04’, '1', '3', '4');

W celu weryfikacji działania bazy przetestowano kilka zapytań SELECT.

- Zapytania o wybrane dane koni sportowych uszeregowane według malejących dat urodzenia

SELECT nazwa, plec_konia, data_urodzenia FROM Konie WHERE przeznaczenie='SPORTOWE' ORDER BY data_urodzenia DESC;

- Zapytanie o nazwę i płeć koni o umaszczeniu siwym

SELECT nazwa, plec_konia FROM Konie NATURAL JOIN Masci_koni WHERE umaszczenie='siwe';

- Zapytanie o nazwę, przeznaczenie i wiek liczony w dniach dla ogierów pochodzących z Polski

SELECT nazwa, przeznaczenie, sysdate-data_urodzenia AS Wiek_konia FROM Konie WHERE plec_konia='OGIER' AND kraj_pochodzenia='Polska';

6. Bibliografia

[1] Materiały dydaktyczne do przedmiotu BDBT

[2] Więzy integralności:

https://education.fandom.com/wiki/PWr_-_Warunki_integralno%C5%9Bci_w_bazie_danych

[3] Stadnina koni w Janowie Podlaskim: https://skjanow.pl/

[4] Stadnina koni nad Wigrami: https://www.nadwigrami.pl/stadnina-koni.html

[5] Stadnina koni ISKRA: http://www.stajniaiskra.pl/art/1/stadnina.html

[6] Indeksy w bazach danych: http://th-www.if.uj.edu.pl/zfs/gora/bazy11/wyklad09.pdf