Projekt 1 - Bazy danych

Adam Nowakowski
Klaudia Stpiczyńska

Listopad 2020

Schronisko dla zwierząt

Spis treści

1 Zakres i cel projektu

Schronisko opiekuje się różnymi zwierzętami są to najczęściej psy i koty. Zajmują się nimi stale zatrudnieni pracownicy. Mogą być oni opiekunami, którzy zajmują się zwierzętami, sprzątają oraz zajmują się pozyskiwaniem zaopatrzenia. Schronisko zatrundnia również weterynarzy.

Baza gromadzi również dane o wolontariuszach, którzy mogą pomogać na dwa sposoby: w codziennej opiece nad zwierzętami oraz przekazując dary.

Informacje o zgromadzonym przez pracowników oraz wolontariuszy zaopatrzeniu (karmie, zabawkach itp.) przechowywane jest w bazie.

Baza danych posiada również informację o wszystkich adoptujących, którzy zdecydowali się przygarnąć zwierzęta.

2 Definicja systemu

2.1 Perspektywy użytkowników

W naszym systemie zdefiniowaliśmy następujące perspektywy użytkownika:

- **Pracownik** - każdy pracownik widzi swoje dane, historię wynagrodzenia oraz dane dotyczące zwierząt, któymi się zajmują
- **Wolontariusz** - widzi swoje dane osobowe, podstawowe informacje o zwierzętach jakimi się zajmuje oraz dary jakie przekazał na schronisko.
- **Adoptujący** - Widzi swoje dane osobowe oraz wszystkie informacje o zwierzętach, jakie adoptował.

2.2 Definicje transakcji

- **Utworzenie biura schroniska** - Podczas założenia biura, rejestrujemy je w bazie.
- **Modyfikacja danych biura (np. nazwy)** - Mamy dostęp do danych określających biuro i możemy je modyfikować.
- **Zatrudnienie pracownika** - Podczas zatrudnienia pracownika, zbieramy o nim informacjie w formularzu i rejestrujemy go w naszej bazie.
- **Modyfikacja obowiązków opiekuna** - Obowiązki opiekuna mogą się zmieniać podczas jego okresu zatrudnienia, więc umożliwiamy modyfikację tego pola w bazie.
- **Wypłata pensji** - Każdą wypłatę rejestrujemy w systemie dla raportów całorocznych.
- **Rejestracja wolontariusza** - Każdego nowego wolontariusza rejestrujemy w naszej bazie.
- **Rejestracja adoptującego** - Podczas procesu adopcji rejestrujemy nowego adoptującego.

- **Przyjęcie nowego zaopatrzenia** - Każde nowe zaopatrzenie musi być zarejestrowane w bazie, aby wiedzieć ile mamy rzeczy.

- **Amortyzacja zaopatrzenia (usunięcie)** - Każde zaopatrzenie się zużywa, więc trzeba pozwolić naszej bazie na łatwe usuwanie tych rzeczy.

- **Zwolnienie pracownika** - Przy zwalnianiu pracownika usuwamy dane o nim.

- **Dostęp do danych o zwierzętach dla adoptującego** - Adoptujący ma możliwość przejrzenia informacji o naszych zwierzętach.

- **Przystąp do procesu adopcji** - Występuje, gdy adoptujący chce zaadoptować kolejne zwierze.

- **Wspomóż darami - wolontariusz** - Gdy wolontariusz przynosi jakieś dary, rejestrujemy je w bazie i oznaczamy je jako przyniesione przez konkretnego pomagającego.

- **Zgłoś się do pomocy nad zwierzakiem** - Wolotariusz może zgłosić się do pomocy nad konkretnym zwerzakiem, np. do jego wyprowadzenia.

- **Dostęp do podopiecznych przez wolontariusza** - Wolontariusz ma dostęp do szczegółowych informacji nt. zwierząt, którym pomaga.

- **Dostęp do aktualnych obowiązków - pracownik** - Pracownik może w łatwy sposób się dowiedzieć jakie aktulnie ma obowiązki.

- **Dostęp do podopiecznych przez pracownika** - Pracownik ma dostęp do szczegółowych informacji nt. zwierząt, którymi się opieuje

3 Model konceptualny

3.1 Definicja zbiorów encji określonych w projekcie (decyzje projektowe)

W naszym projekcie zdefiniowaliśmy następujące encje:

- Schronisko
- Pracownik ze specjalizacjami weterynarz i opiekun
- Wolontariusz
- Zwierzę
- Adoptujący
- Zaopatrzenie

Główną encją jest "Schronisko". Jedną z najważniejszych czynności, było zbieranie informacji o zwierzętach, które znajdują się w schronisku, dlatego stworzyliśmy encję zwierzę.

Ponieważ w opiece nad zwierzętami zajmują zarówno zatrudnione na stałe osoby jak i ludzie, którzy chcą pomóc, podzieliliśmy ich na dwie encję: pracownik i wolontariusz. Dokonaliśmy podziału encji pracownik na specjalizacje: weterynarz i opiekun, ponieważ w przypadku weterynarzy gromadzimy takie dany jak numer licencji weterynaryjnej, a w przypadku opiekunów musimy ustalić obowiązki, jakie będą pełnić.

Kolejnym typem osoby, która może wesprzeć schronisko, jest ktoś, kto przygarnie do siebie zwierzę. W tym momencie jest on rejetrowany w systemie, stąd podjęliśmy decyzję o wprowadzeniu encji Adoptujący.

Potrzebne również były informacje o darach przekazanych przez innych oraz o aktualnie dostępnych zasobach w schronisku. Stąd decyzja o wprowadzeniu encji zaopatrzenie.

3.2 Ustalenie związków między encjami i ich typów

Pomiędzy encjami zdefiniowanymi w punkcie 3.1 zdefiniowaliśmy następujące związki:

- **Schronisko - Pracownik** - jest to związek typu jeden do wielu. Skchronisko może **zatrudniać** wiele pracowników, a w skrajnym przypadku zero (przy jego założeniu). Pracownik **jest zatrudniany** i od razu rejestrowany w bazie, dlatego może być przydzielony tylko do jednego schroniska.

- **Schronisko - Zwierzę** - schronisko może być **zamieszkiwane** przez wiele zwierząt, a w skrajnym przypadku przez zero. Zwierzę, które rejestrujemy może **zamieszkiwać** tylko jedno, nasze schronisko.

- **Schronisko - Wolontariusz** - jest to sytuacja analogiczna do dwóch powyższych. Schronisko **jest wspierane** przez wielu wolontariuszy, a zarejestrowany w naszej bazie wolontariusz może **wspierać** tylko nasze schronisko.

- **Schronisko - Zaopatrzenie** - jest to związek typu jeden do wielu. Schronisko może **posiadać** wiele sztuk zaopatrzenia różnego rodzaju. A zaopatrzenie **jest posiadane** i przypisane do naszego schroniska.

- **Pracownik - Zwierzę** - w tym przypadku mówimy o związku typu wiele do wielu. Pracownik ma przypisanych swoich podopiecznych i **opiekuje się** wieloma zwierzętami. W skrajnym przypadku może nie opiekować się żadnym zwierzęciem, gdyż może to nie być w jego obowiązkach. Zwierzę ma przypisanego co najmniej jednego opiekuna, lecz na co dzień **będzie w opiece** kilku pracowników, np. w systemie zmianowym.

- **Pracownik - Zaopatrzenie** - kolejny związek typu jeden do wielu. Pracownik może **pozyskać** zaopatrzenie (zero lub wiele). Zaopatrzenie **może być pozyskane** przez jednego pracownika lub przez żadnego (w przypadku, gdy jest darowane przez wolontariusza).

- **Wolontariusz - Zwierzę** - ten przypadek relacji jest analogiczny do pracownik - zwierzę. Jedyną różnicą jest to, że w skrajnym przypadku zarejestrowany wolontariusz nie będzie **pomagał** (w sposób fizyczny) żadnym zwierzęciem - będzie tylko darczyńcą.

- **Zwierzę - Zaopatrzenie** - zwierzę może **użytkować** wiele sztuk zaopatrzenia (np. obrożę, miskę, smycz). Zaopatrzenie z kolei **może być użytkowane** przez kilka zwierząt (np. smycz).

- **Zwierzę - Adoptujący** - zwierzę **może być adoptowane** przez jednego adoptującego, a w mniej optymistycznym przypadku przez żadnego - będzie szukało domu. Za to adoptującego rejestrujemy

dopiero w chwili **adopcji** podopiecznego. Więc minimalnie może adoptować jednego zwierzaka a maksymalnie wiele.

- **Adoptujący - Schronisko** - Jak było wspomniane powyzej, adoptujący **jest rejestrowany** w chwili adopcji zwierzaka. I jest zarejestrowany minimum do jednego a maksymalnie do wielu schronisk. Schronisko może **rejestrować** wielu adoptujących, a w skrajnym wypadku zero.

3.3 Określenie atrybutów i ich dziedzin

Do każdej encji zdefiniowaliśmy atrybuty opisujące je. Poniżej wszystkie zostały opisane.

- Schronisko
  1. Nr_schroniska - jest to klucz główny encji Schronisko. Zdecydowaliśmy się na wybór klucza sztucznego. Typ danych jaki wybraliśmy to Smallint. Jako klucz główny jest to pole unikalne i niemogące być puste.
  2. Nazwa - nazwa nadana schronisku. Typ danych zdefiniowaliśmy jako Varchar(30) oraz oznaczyliśmy to pole jako obowiązkowe.
  3. Adres - w tym atrybucie zapisywany jest adres schroniska, czyli takie dane jak: miasto, ulica, numer domu, numer mieszkania oraz kod pocztowy. Z powodu dużej ilości informacji przyjęliśmy typ Varchar(300). Również to pole jest obowiązkowe.
  4. Założyciel - atrybut pozwalający na zapisanie imion i nazwisk założycieli schroniska. Ponieważ założycieli mogło być kilku typ, jaki przyjęliśmy to Varchar(150). Również to pole ma oznaczeniea "Mandatory".
  5. Data założenia - jest to data założenia schroniska, dlatego wybraliśmy typ jako Date. Również i to pole zostało oznaczone jako obowiązkowe.
- Pracownik
  1. Nr_pracownika - jest to klucz główny encji Pracownik. Wybraliśmy klucza sztuczny. Typ danych jaki wybraliśmy to Integer. Jako klucz główny jest to pole unikalne i niemogące być puste.
  2. Imię - jest to imię pracownika, typ danych to Varchar(20). Również jest to pole obowiązkowe.
  3. Nazwisko - naziwsko pracownika, z uwagi na nazwiska dwuczłonowe zdecydowaliśmy się na typ Varchar(40). Pole jest oznaczone jako "Mandatory".
  4. Płeć - przy wyborze płci zdefiniowaliśmy regułę, która pozwala na wybór z dwóch opcji: 'K' oraz 'M', pole to jest obowiązkowe.
  5. Data urodzenia - pole typu Date opisujące datę urodzenia pracownika. Pracownik podczas zatrudnienia musi podać tę informację.
  6. Data zatrudnienia - pole analgiczne jak data urodzenia, opisujące dzień zatrudnienia pracownika.
  7. Pensja - pole typu Float(2), rozumiemy to jako przechowywanie informacji o wypłacanych co miesiąc wynagrodzeniach. Dlatego jest to pole nieobowiązkowe, ponieważ po zatrudnieniu pracownika może być czas, kiedy nie otrzymał jeszcze wynagrodzenia.

8. Pesel - pole typu Integer. Oznaczyliśmy to pole jako nieobowiązkowe, ponieważ dopuszczamy zatrudnianie osób z zagranicy, które mogą nie posiadać numeru pesel.

9. Adres - w tym atrybucie zapisywany jest adres pracownika, czyli takie dane jak: miasto, ulica, numer domu, numer mieszkania oraz kod pocztowy. Z powodu dużej ilości informacji przyjęliśmy typ Varchar(300). Pole jest obowiązkowe.

- Weterynarz

1. Nr_licencji- oprócz atrybutów opisujących wszystkich pracowników, weterynarz posiada ten atrybut, w którym przechowujemy numer licencji weterynaryjnej. Jest to obowiąkowe pole typu Integer.

2. $\mathrm{Data}_{z}akonczenia_{s}tudiow - obowizkowepoletypuDate, mwiceotym, kiedyweterynarzzakoczystudia$
$Stanowisko{-}obowizkowepolezezdefiniowanregu, wktrejdowyborujestWETERYNARZlubTECHNIK(wskrcieo$

3. Opiekun

1. Wymiar_godzin - analogicznie jak w przypadku weterynarza, opiekuna opisują takie same atrybuty jak wszystkich pracowników, a także dodatkowe atrybuty. Wymiar godzin jest to oowiązkowe pole typu Smallint, które określa, ile godzin w tygodniu pracuje opiekun.

2. Obowiązki - określa jakie obowiązki w schronisku pełni opiekun. Typ tego atrybutu to Varchar(100), również jest oznaczone jako obowiązkowe.

3. Niekaralnosc - obowiązkowe pole, w którym można wybrać jedną z opcji - TAK lub NIE, mówiące o tym, czy opiekun posiada zaświadzenie o niekaralności.

- Wolontariusz

1. Nr_wolontariusza - jest to klucz główny encji Pracownik. Wybraliśmy klucza sztuczny. Typ danych jaki wybraliśmy to Integer. Jako klucz główny jest to pole unikalne i niemogące być puste.

2. Imię - jest to imię wolontariusza, typ danych to Varchar(20). Również jest to pole obowiązkowe.

3. Nazwisko - analogicznie jak w przypadku pracownika ze względu na możliwość podwójnych nazwisk wybraliśmy je jako pole typu Varchar(40) i oznaczyliśmy jako "Mandatory".

4. Adres - w tym atrybucie zapisywany jest adres wolontariusza, czyli takie dane jak: miasto, ulica, numer domu, numer mieszkania oraz kod pocztowy. Z powodu dużej ilości informacji przyjęliśmy typ Varchar(300). Pole jest obowiązkowe.

5. Pesel - pole typu Integer. Na wypadek gdyby wolontariuszem chciał zostać ktoś z zagranicy, pole to nie jest obowiązkowe.

6. Rola - jest to pole określające, czy dany wolontriusz zajmuje się opieką nad zwierzętami, czy jest darczyńcą - do wybory są opcje O - opiekun, D -darczyńca, OBA - obie role.

- Zwierzę

1. Nr_zwierzęcia - jest to klucz główny encji Zwierzę. Wybraliśmy klucza sztuczny. Typ danych jaki wybraliśmy to Integer. Jako klucz główny jest to pole unikalne i niemogące być puste.

2. Gatunek - w tym polu zdefiniowaliśmy regułę pozwalającą wybrać z trzech opcji: 'PIES', 'KOT', 'INNE', ponieważ psy i koty są najczęściej spotykanymi zwierzętami w schronisku. Inne zwierzęta (np. egzotyczne) są na tyle rzadko spotykane, że oznaczamy je opją 'INNE'. Pole to jest obowiązkowe.

3. Rasa - opisuje jakiej rasy jest dane zwierzę. Jest to obowiązkowe pole typu Varchar(30).

4. Imię - imię nadane zwierzęciu. Ponieważ zwierzę może być zarejestrowane do schroniska, zanim zostanie nadane mu imię, nie jest to pole obowiązkowe. Typ definiowany w tym atrybucie jest to Varchar(20).

5. Rok urodzenia - rok urodzenia zwierzęcia, typu Integer. Z uwagi na to, że może się zdarzyć, że przygarniając zwierzę, nie będziemy znali czasu jego urodzenia, pole to jest nieobowiązkowe.

- Adoptujący

1. Nr_adoptujacego - jest to klucz główny encji Adoptujący. Wybraliśmy klucza sztuczny. Typ danych jaki wybraliśmy to Integer. Jako klucz główny jest to pole unikalne i niemogące być puste.

2. Imię - imię osoby, która adoptuje zwierzę. Atrybut typu Varchar(20), pole obowiązkowe.

3. Nazwisko - nazwisko osoby adoptującej, analogicznie do atrybutów innych encji powiązanych z nazwiskiem, wybraliśmy ten atrybut jak obowiązkowy typu Varchar(40).

4. Adres - w tym atrybucie zapisywany jest adres adoptującego, czyli takie dane jak: miasto, ulica, numer domu, numer mieszkania oraz kod pocztowy. Z powodu dużej ilości informacji przyjęliśmy typ Varchar(300). Pole jest obowiązkowe.

5. Pesel - z uwagi na możliwość adoptowania zwierząt przez obcokrajowców, jest to pole nieobowiązkowe typu Integer.

- Zaopatrzenie

1. Nr_zaopatrzenia - jest to klucz główny encji Adoptujący. Wybraliśmy klucza sztuczny. Typ danych jaki wybraliśmy to Integer. Jako klucz główny jest to pole unikalne i niemogące być puste.

2. Rodzaj - w tym przypadku zdefiniowaliśmy regułę, która daje możliwość do wyboru z: 'KARMA', 'ZABAWKA', 'SMYCZ','MISKA', jest to pole obowiązkowe.

3. Nazwa - jest to nazwa danego zaopatrzenia np. karma dla kota, jest to obowiązkowe pole typu Varchar(20).

4. Ilość - ilość danego zaopatrzenia np. 5 kilo, z tego względu jest to pole typu Varchar(20), oznaczone jest jako "Mandatory".

5. Dostępność - zdefiniowaliśmy regułę, która pozwala wybrać dstępność z 'TAK' lub 'NIE'. To pole wyjaśnia, czy dany produkt jest jeszcze dostępny do użycia. Atrybut ten jest obowiązkowy.

3.4 Dodatkowe reguły integralnosciowe (reguły biznesowe)

Aby nasz model był jak najbardziej przybliżony do rzeczywistości, potrzebne są dodatkowe ograniczenia na poszczególne atrybuty danych encji.

- **Schronisko**
  - może być wielu założycieli
- **Pracownik**
  - pensja musi być wartością większą od zera i może być polem pustym, gdyż jest wpisywana dopiero, jak pracownik ją otrzyma. Wartość w tym atrybucie może się zmieniać na przestrzeni miesięcy (podwyżka / awans)
  - PESEL nie jest obligatoryjnym atrybutem, może pozostać puste, w przypadku, gdy naszym pracownikiem jest obcokrajowiec
- **Adoptujący i wolontariusz**
  - PESEL nie jest obligatoryjnym atrybutem, może pozostać puste, w przypadku, gdy naszym pracownikiem jest obcokrajowiec
- **Zaopatrzenie**
  - Ilość i dostępność nie może być mniejsza od zera

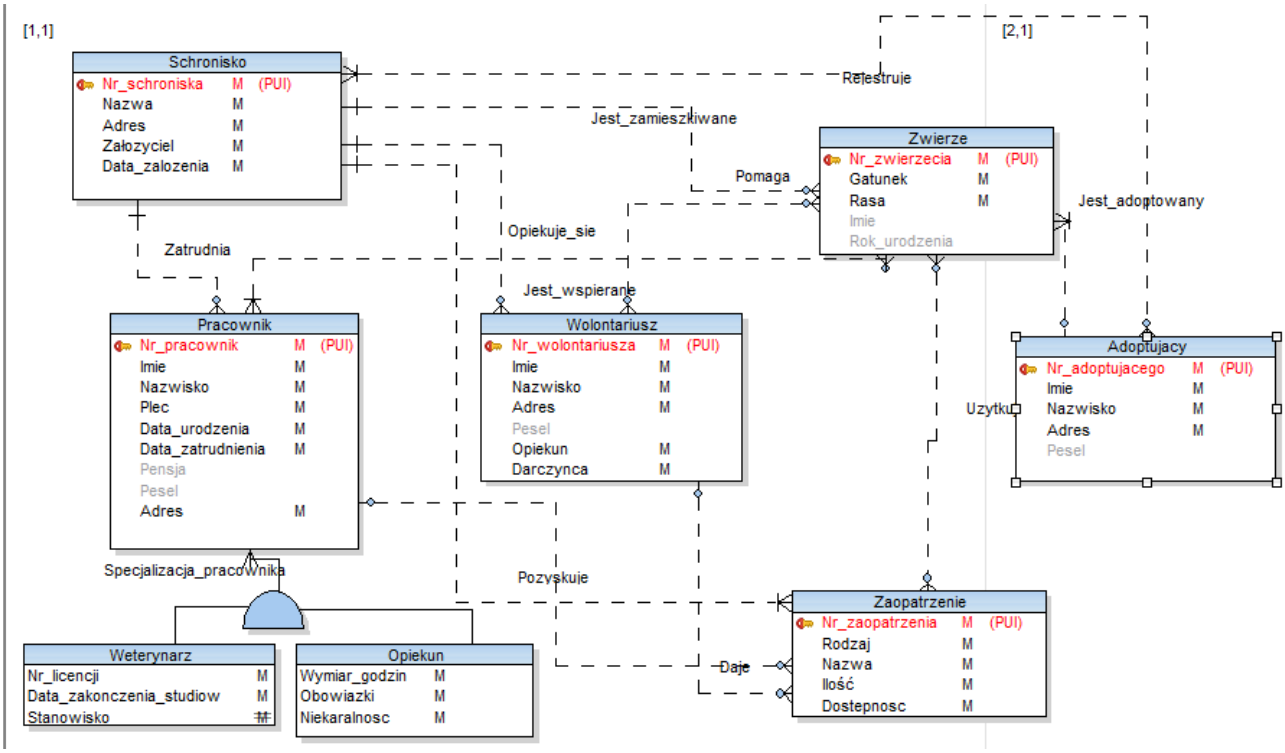
3.5 Klucze kandydujące i główne (decyzje projektowe)

Zdecydowaliśmy się podczas tworzenia modelu konceptualnego, że każdy klucz główny będzie kluczem sztucznym. W encjach moglibyśm zdefiniować inne klucze, składające się z grup atrybutów. Zdecydowaliśmy się jednak na inne rozwiązanie, żeby zapewnić sobie klucze proste. Takie rozwiązanie ma szereg zalet. Jedną z nich jest fakt, że jeżeli osiągnęliśmy pierwsą postać normalną i wszystkie klucze są kluczami prostymi, nasz projekt osiągnął również drugą postać normalną. Na przykład:

- W encji schronisko kluczem kandydującym może być jego nazwa. Przy zakładaniu nowego schroniska powinniśmy wtedy wymagać unikalności nazwy. Nie jest to dobra praktyka, gdyż nie wiemy co zamierza zrobić w takiej sytuacji nasz klient. Powinniśmy dać mu swobodę w nazywaniu swoich placówek.
- W encjach związanych z ludźmi kluczem kandydującym może być PESEL. Jest to unikalny numer nadawany w Polsce. Jestety obcokrajowcy nie mają przypisanego numeru PESEL. To pole musi więc pozostać nieobowiazkowe, więc nie może być kluczem głównym.

3.6 Schemat ER na poziomie konceptualnym

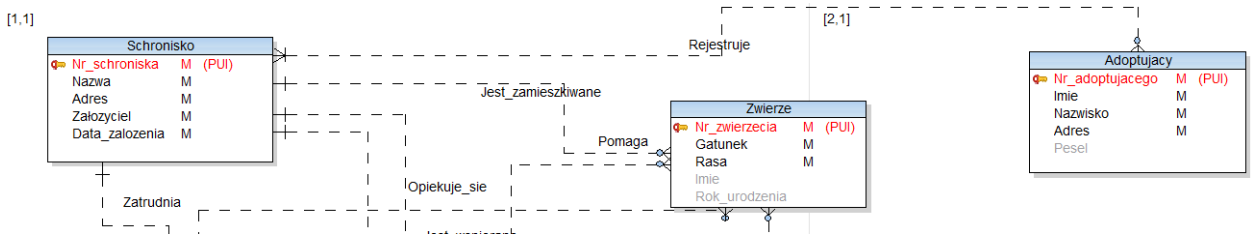
Schemat ER na poziomie konceptualnym jest przedstawiony na rysunku 1:



Rysunek 1: Schemat ER na poziomie konceptualnym

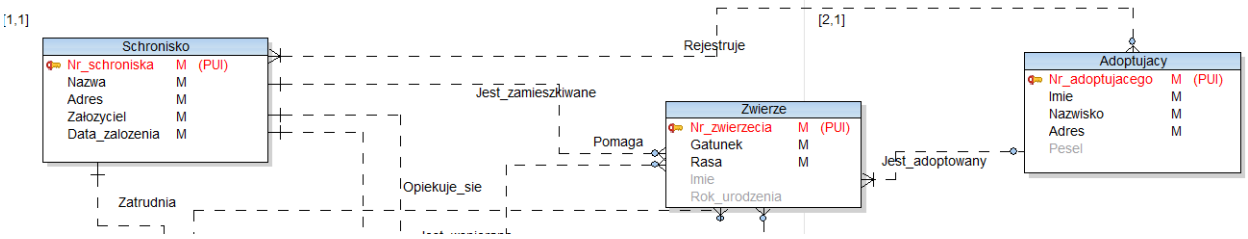
3.7 Problem pułapek szczelinowych i wachlarzowych – analiza i przykłady

- **Pułapka wachlarzowa** - występuje w sytuacji, gdy model przedstawia związek pomiędzy pewnymi zbiorami encji, ale wynikające z tego ścieżki pomiędzy wystąpieniami encji nie są jednoznaczne. Pułapka występowałaby w następującej sytuacji: Mamy połączenie pomiędzy encjami Adoptujący - Schronisko i Schronisko - Zwierze. W takim wypadku mając konkretnego adoptującego nie jesteśmy w stanie zidentyfikować konkretnego zwierzęcia.



Rysunek 2: Przykład pułapki wachlarzowej

Rozwiązaniem tego problemu może być wstawienie relacji pomiędzy Zwierzęciem a Adoptującym



Rysunek 3: Rozwiązanie pułapki wachlarzowej

- **Pułapka szczelinowa** - wystąpiła u nas podczas rozwarzań na temat encji Zaopatrzenie. W ogólnym przypadku zaopatrzenie jest pozyskiwane przez pracowników i wolontariuszy i użytkowane przez zwierzęta. W skrajnym przypadku może okazać się, że podstawowe zaopatrzenie nie jest połączone z żadnym z powyższych encji. Rozwiązaliśmy to bezpośrednim połączeniem Schroniska z Zaopatrzeniem.

4 Model logiczny

4.1 Charakterystyka modelu relacyjnego

Model relacyjny jest to model organizacji danych bazujący na matematycznej teorii mnogości. W tym modelu dane grupowane są w relacje (tabele), w których każda ma określone atrybuty (są to kolumny każdej tabeli). Można również definiować pojęcie krotki, czyli wiersza relacji. Podczas definiowania związków między encjami możemy dodawać klucze obce do relacji, które pozwalają nam się odnosić do innej tabeli.

Takie ujęcie modelu pozwala na dużą niezależność danych, czyli modyfikacja struktur danych, nie powinna mieć wpływu na wykorzystujące je aplikacje. Model relacyjny pozwala również na uniknięcie redundancji danych.

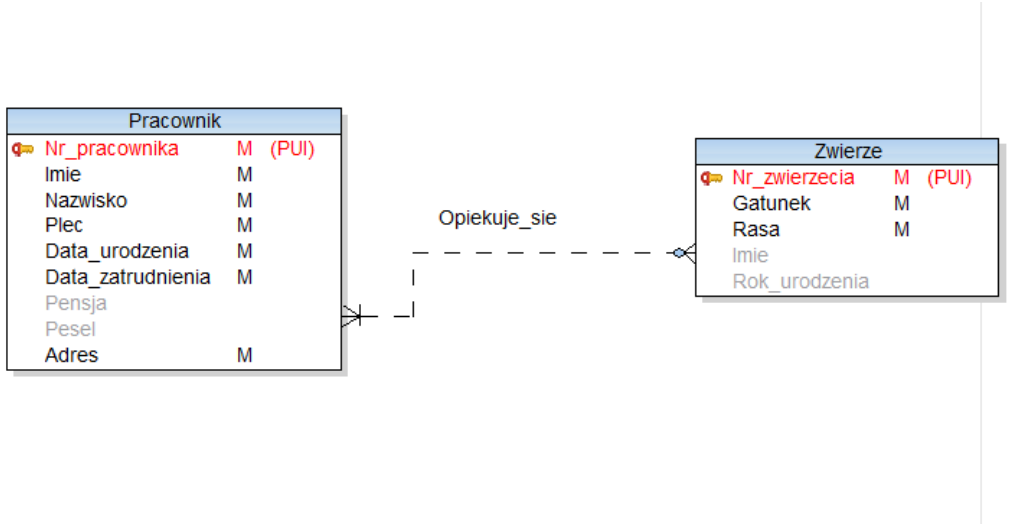
4.2 Usunięcie właściwości niekompatybilnych z modelem relacyjnym - przykłady

W tym momencie zmieniliśmy nazwy encji z liczb pojedynczych na mnogie np.:

Wolontariusz zmienione na Wolontariusze

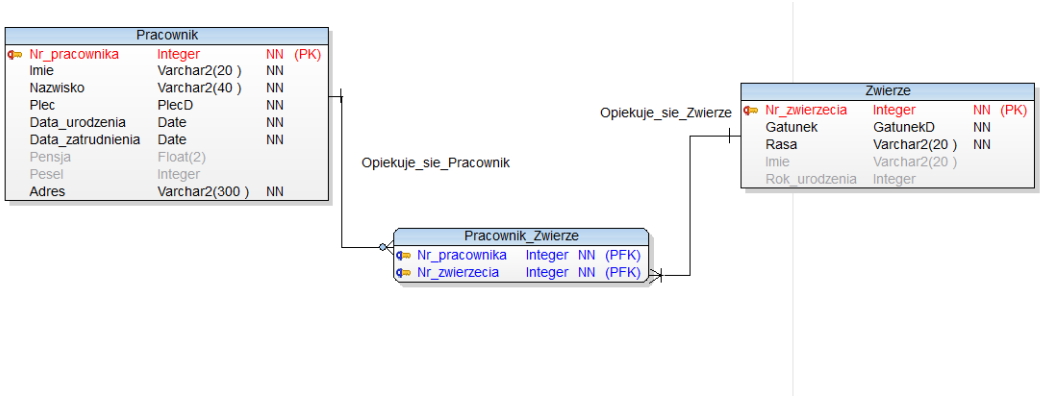
Najważniejsze w tym punkcie było zamienienie wszystkih relacji wiele do wielu na tablice łączące, ponieważ w modelu relacyjnym nie mogą znajdować się takie relacje.

W ten sposób np. następująca relacja pomiędzy dwoma encjami:



Rysunek 4: Fragment modelu na poziomie konceptualnym

Została zmieniona na:



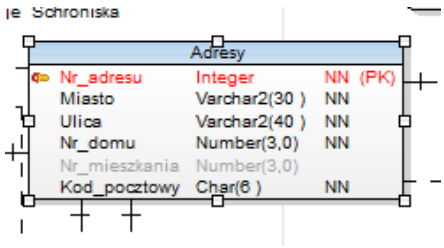
Rysunek 5: Fragment modelu na poziomie logicznym

4.3 Proces normalizacji – analiza i przykłady

Podczas procesu normalizacji znaleźliśmy następujące miejsca, które nie spełniały trzeciej postaci normalnej:

4.3.1 Pole Adresa w encji Schroniska

Pole to było polem wielowartościowym, dlatego stworzyliśmy następującą relację, opisującą Adres:

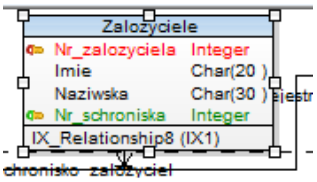


Rysunek 6: Relacja Adresy

Pole adres w encji Schroniska zastąpiliśmy kluczem obcym relacji Adresy. Ponieważ taka sama sytuacja występowała w innych encjach (Pracownicy, Adoptujący, Wolontariusze), postąpiliśmy tam analogicznie, używając wyżej przedstawionej relacji Adresy.

4.3.2 Pole Założyciel w relacji Schroniska

Ponieważ dopuszczamy możliwość, że był więcej niż jeden założyciel schroniska, to pole jest potencjalnie wielowartościowe. Dlatego zdecydowaliśmy się na stworzenie kolejnej relacji Założyciele, przedstawionej na rysnku 7.

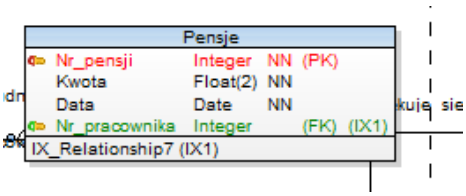


Rysunek 7: Relacja Zalozyciele

Założyciela, oprócz klucza głównego, opisują takie atrybuty jak imię, nazwisko oraz numer schroniska jakie założył, podane jako klucz obcy z relacji schroniska.

4.3.3 Pole Pensja w relacji Pracownicy

W tym atrybucie chcieliśmy przechowywać informację o wypłacanych wynagrodzeniach na przestrzeni całęgo zatrudnienia. Ponieważ byłaby to powtarzająca się grupa danych, zdecydowaliśmy się na stworzenie kolejnej relacji Pensje. Przedstawiona jest na rysunku 8.



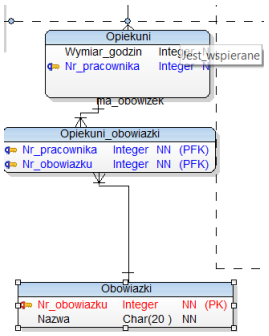
Rysunek 8: Relacja Pensje

W tej relacji oprócz sztucznego klucza głównego mamy takie atrybuty jak kwota oraz data wypłacenia pensji. Również jako klucz obcy został dodany numer pracownika, któremu została wypłacona pensja.

4.3.4 Pole Obowiązki w relacji Opiekuni

Każdy opiekun może wykonywać więcej niż jeden obowiązek, dlatego pole to byłoby wielowartościowe. Zdecydowaliśmy się na stworzenie relacji słownikowej, która zawiera klucz główny oraz nazwę obowiązku. Ponieważ

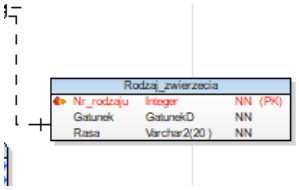
każdy obowiązek może pełnić wielu pracowników, zdecydowaliśmy się na stworzenie tablicy łączącej. Ostatecznie relacje te wyglądały następująco:



Rysunek 9: Relacje Obowiązki

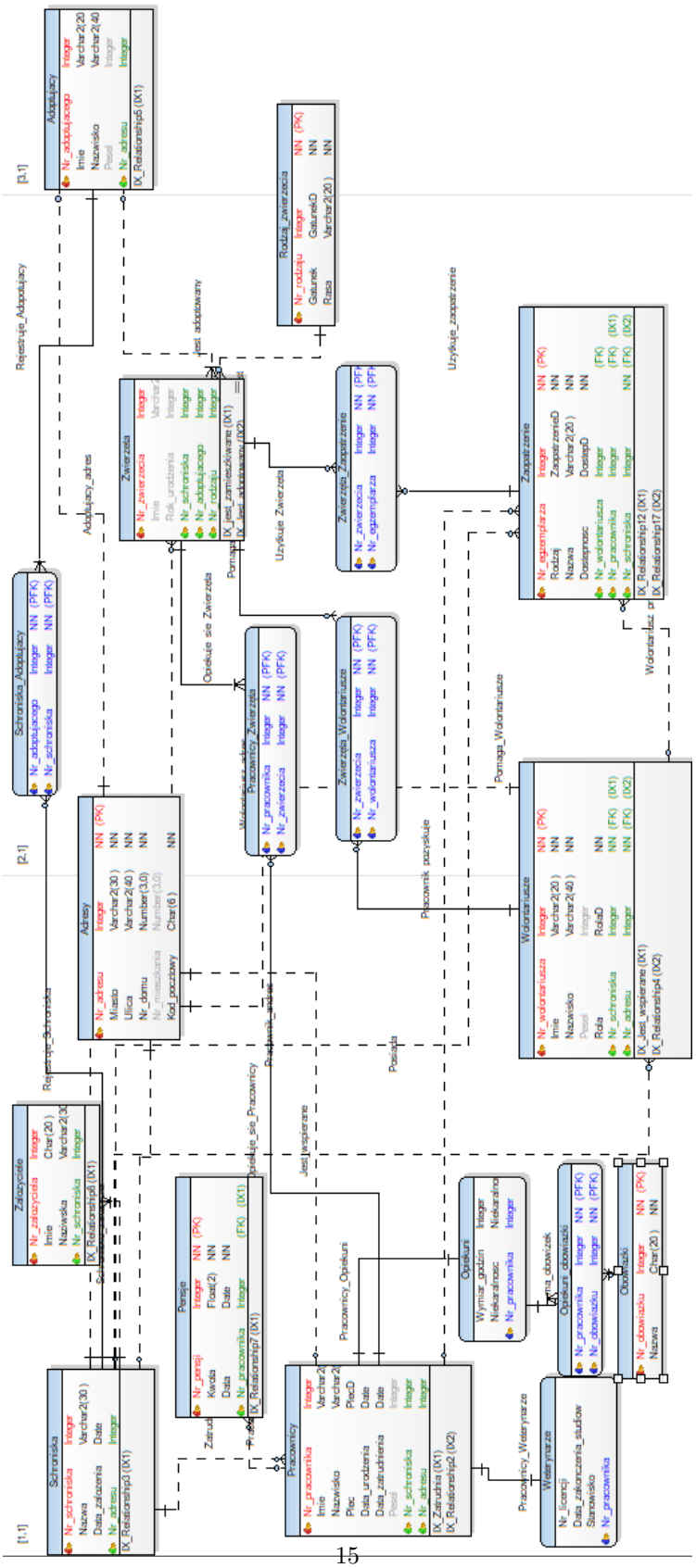
4.3.5 Relacja słownikowa - rodzaj zwierzęcia

W encji rodzaj zwierzęcia występowały powtarzające się grupy. Na przykład kot syjamski mógł występować wiele razy. Poztanowiliśmy dodać relację słownikową określającą rodzaj zwierzęcia. Ta relacja ma klucz główny, który jest kluczem obcym dla każdej encji zwierzę.



Rysunek 10: Relacja rodzaj zwierzęcia

4.4 Schemat ER na poziomie modelu logicznego



Rysunek 11: Schemat ER na poziomie logicznym

4.5 Więzy integralności

Aby zachować spójność danych w przypadku ich modyfikacji stosujemy tzw. więzy integralności. Są to na przykład:

- Stosowanie kluczy głównych - gwarantują unikalność wartości w kolumnie
- Stosowanie kluczy obcych - gwarantują, iż rekord w tabeli podrzędnej zawsze będzie miał swojego odpowiednika w tabeli nadrzędnej
- Stosowanie wartości NOT NULL

4.6 Proces denormalizacji – analiza i przykłady

Denormalizacja jest to proces celowego wprowadzenia redundancji danych w naszej bazie. Jest to niejako kompromis pomiędzy wprowadzeniem bazy w poszczególne postacie normalne, a np. wydajnością tej bazy. Bierzemy pod analizę następujące przejawy celowej denormalizacji:

- **Redukcja liczby złączeń poprzez powielanie atrybutów, które nie należą do kluczy w związkach 1:N**
  Taką operację można zastosować np. w związku Schronisko - Pracownik. Żeby mieć bezpośredni dostęp z poziomu encji pracownika, w jakim schronisku pracuje (nazwa). Niesie to za sobą poważne konsekwencje. Mianowicie jak nazwa biura się zmieni, trzeba pamiętać, żeby zaktualizować tą nazwę przy każdym pracowniku.
  Nie widziedliśmy potrzeby stosowania powyższej praktyki.
- **Redukcja liczby złączeń poprzez powielanie atrybutów kluczy obcych w związkach 1:N**
  Taką operację można zastosować np. w związkach Schronisko - Pracownik - Zaopatrzenie. Jednocześnie chcemy, aby z poziomu encji Schroniska mieć dostęp do całego zaopatrzenia, a także z poziomu Pracownika wiedzieć, jakie zaopatrzenie zostało przez niego pozyskane. Występuje tutaj cykl, czyli do encji Zaopatrzenia z encji Biura można dojść wieloma ścieżkami.
  W modelu naszej bazy zastosowaliśmy właśnie to rozwiązanie, gdyż wydaje nam się ono bardzo sensowne. Jest to nam też potrzebne przy likwidacji pułapki szczelinowej.

5 Faza fizyczna

5.1 Projekt transakcji i weryfikacja ich wykonalności

Obszar	Transakcja	Zasób	Czy możliwe?	Uwagi
Biuro	Utworzenie biura schroniska	Adresy, Założyciele	Tak	Musi istnieć adres i założyciele
	Modyfikacja danych biura (np. nazwy)	Biura	Tak	
Administracja	Zatrudnienie pracownika	Pracownicy	Tak	Musi istnieć adres
	Modyfikacja obowiązków opiekuna	Opiekuni, Obowiązki	Tak	Musi istnieć dany obowiązek
	Wypłata pensji	Pracownicy, Pensje	Tak	-
	Rejestracja wolontariusza	Wolontariusze	Tak	Musi istnieć adres
	Rejestracja adoptującego	Adoptujący, Zwierzęta	Tak	Musi istnieć zwierzę, które chcemy adoptować
	Przyjęcie nowego zaopatrzenia	Zaopatrzenia, Wolontariusze / Pracownicy	Tak	-
	Amortyzacja zaopatrzenia (usunięcie)	Zaopatrzenie	Tak	-
	Zwolnienie pracownika	Pracownicy	Tak	-
Panel Adoptującego	Dostęp do danych o zwierzętach	Zwierzęta	Tak	-
	Przystąp do procesu adopcji	Zwierzęta	Tak	-
Panel Wolontariusza	Wspomóż darami	Zaopatrzenie	Tak	Musi istnieć dane zaopatrzenie
	Zgłoś się do pomocy nad zwierzakiem	Zwierzęta	Tak	Musi istnieć dane zwierze
	Dostęp do podopiecznych	Zwierzęta	Tak	-
Panel Pracownika	Dostęp do aktualnych obowiązków	Obowiązki	Tak	-
	Dostęp do podopiecznych	Zwierzęta	Tak	-
	Dostęp do podopiecznych	Pensje	Tak	-

5.2 Strojenie bazy danych – dobór indeksów

Indeksy jest to struktura, ułatwiająca wyszukiwanie danych. POdczas genracji kodu za pomocą oprogramowania Toad Data Modeler skrypt DDL zawiera indeksy, mogące występować najczęściej. Przykładem w kodzie, gdzie tworzone są indeksy jest:

```
CREATE INDEX IX_Zatrudnia ON Pracownicy (Nr_schroniska)
/

CREATE INDEX IX_Relationship2 ON Pracownicy (Nr_adresu)
/
```

5.3 Skrypt SQL zakładający bazę danych

```
/*
Created: 14.11.2020
Modified: 20.11.2020
Model: Schronisko
Database: Oracle 12c
*/


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

CREATE SEQUENCE Schroniska_seq1
 INCREMENT BY 1
 START WITH 1
 NOMAXVALUE
 NOMINVALUE
 CACHE 20
/

CREATE SEQUENCE Zalozyciel_seq1
 INCREMENT BY 1
 START WITH 1
 NOMAXVALUE
 NOMINVALUE
 CACHE 20
/

CREATE SEQUENCE Adres_seq1
 INCREMENT BY 1
 START WITH 1
 NOMAXVALUE
 NOMINVALUE
 CACHE 20
/

CREATE SEQUENCE Zwierzeta_seq1
 INCREMENT BY 1
 START WITH 1
 NOMAXVALUE
 NOMINVALUE
 CACHE 20
```

```
/

CREATE SEQUENCE Adoptujacy_seq1
 INCREMENT BY 1
 START WITH 1
 NOMAXVALUE
 NOMINVALUE
 CACHE 20
/

CREATE SEQUENCE Pensje_seq1
 INCREMENT BY 1
 START WITH 1
 NOMAXVALUE
 NOMINVALUE
 CACHE 20
/

CREATE SEQUENCE Pracownicy_seq1
 INCREMENT BY 1
 START WITH 1
 NOMAXVALUE
 NOMINVALUE
 CACHE 20
/

CREATE SEQUENCE Obowiazki_seq1
 INCREMENT BY 1
 START WITH 1
 NOMAXVALUE
 NOMINVALUE
 CACHE 20
/

CREATE SEQUENCE Wolontariusze_seq1
 INCREMENT BY 1
 START WITH 1
 NOMAXVALUE
 NOMINVALUE
 CACHE 20
/

CREATE SEQUENCE Zaopatrzenie_seq1
 INCREMENT BY 1
 START WITH 1
 NOMAXVALUE
 NOMINVALUE
 CACHE 20
/

CREATE SEQUENCE Egzemplarze_seq1
 INCREMENT BY 1
 START WITH 1
 NOMAXVALUE
```

```
 NOMINVALUE
 CACHE 20
/

CREATE SEQUENCE Rodzaj_seq1
 INCREMENT BY 1
 START WITH 1
 NOMAXVALUE
 NOMINVALUE
 CACHE 20
/

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

-- Table Schroniska

CREATE TABLE Schroniska(
  Nr_schroniska Integer NOT NULL,
  Nazwa Varchar2(30 ) NOT NULL,
  Data_zalozenia Date NOT NULL,
  Nr_adresu Integer NOT NULL
)
/

-- Create indexes for table Schroniska

CREATE INDEX IX_Relationship3 ON Schroniska (Nr_adresu)
/

-- Add keys for table Schroniska

ALTER TABLE Schroniska ADD CONSTRAINT Unique_Identifier1
PRIMARY KEY (Nr_schroniska)
/

-- Table Pracownicy

CREATE TABLE Pracownicy(
  Nr_pracownika Integer NOT NULL,
  Imie Varchar2(20 ) NOT NULL,
  Nazwisko Varchar2(40 ) NOT NULL,
  Plec Char(1 ) NOT NULL
        CHECK (Plec IN ('K','M')),
  Data_urodzenia Date NOT NULL,
  Data_zatrudnienia Date NOT NULL,
  Pesel Integer,
  Nr_schroniska Integer NOT NULL,
  Nr_adresu Integer NOT NULL
)
/

-- Create indexes for table Pracownicy

CREATE INDEX IX_Zatrudnia ON Pracownicy (Nr_schroniska)
```

```
/

CREATE INDEX IX_Relationship2 ON Pracownicy (Nr_adresu)
/

-- Add keys for table Pracownicy

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

-- Table Zwierzeta

CREATE TABLE Zwierzeta(
  Nr_zwierzecia Integer NOT NULL,
  Imie Varchar2(20 ),
  Rok_urodzenia Integer,
  Nr_schroniska Integer NOT NULL,
  Nr_adoptujacego Integer,
  Nr_rodzaju Integer NOT NULL
)
/

-- Create indexes for table Zwierzeta

CREATE INDEX IX_jest_zamieszkiwane ON Zwierzeta (Nr_schroniska)
/

CREATE INDEX IX_Jest_adoptowany ON Zwierzeta (Nr_adoptujacego)
/

CREATE INDEX IX_Relationship21 ON Zwierzeta (Nr_rodzaju)
/

-- Add keys for table Zwierzeta

ALTER TABLE Zwierzeta ADD CONSTRAINT Unique_Identifier3 PRIMARY KEY (Nr_zwierzecia)
/

-- Table Adoptujacy

CREATE TABLE Adoptujacy(
  Nr_adoptujacego Integer NOT NULL,
  Imie Varchar2(20 ) NOT NULL,
  Nazwisko Varchar2(40 ) NOT NULL,
  Pesel Integer,
  Nr_adresu Integer NOT NULL
)
/

-- Create indexes for table Adoptujacy

CREATE INDEX IX_Relationship5 ON Adoptujacy (Nr_adresu)
/
```

```
-- Add keys for table Adoptujacy

ALTER TABLE Adoptujacy ADD CONSTRAINT Unique_Identifier6 PRIMARY KEY (Nr_adoptujacego)
/

-- Table Wolontariusze

CREATE TABLE Wolontariusze(
  Nr_wolontariusza Integer NOT NULL,
  Imie Varchar2(20 ) NOT NULL,
  Nazwisko Varchar2(40 ) NOT NULL,
  Pesel Integer,
  Rola Varchar2(3 ) NOT NULL
        CHECK (Rola IN ('O','D', 'OBA')),
  Nr_schroniska Integer NOT NULL,
  Nr_adresu Integer NOT NULL
)
/

-- Create indexes for table Wolontariusze

CREATE INDEX IX_Jest_wspierane ON Wolontariusze (Nr_schroniska)
/

CREATE INDEX IX_Relationship4 ON Wolontariusze (Nr_adresu)
/

-- Add keys for table Wolontariusze

ALTER TABLE Wolontariusze ADD CONSTRAINT Unique_Identifier7 PRIMARY KEY (Nr_wolontariusza)
/

-- Table Weterynarze

CREATE TABLE Weterynarze(
  Nr_licencji Integer NOT NULL,
  Data_zakonczenia_studiow Date NOT NULL,
  Stanowisko Varchar2(10 ) NOT NULL
        CHECK (Stanowisko IN ('WETERYNARZ', 'TECHNIK')),
  Nr_pracownika Integer NOT NULL
)
/

-- Add keys for table Weterynarze

ALTER TABLE Weterynarze ADD CONSTRAINT Unique_Identifier9
PRIMARY KEY (Nr_pracownika)
/

-- Table Opiekuni

CREATE TABLE Opiekuni(
  Wymiar_godzin Integer NOT NULL,
  Niekaralnosc Char(3 ) NOT NULL
```

```
        CHECK (Niekaralnosc IN ('TAK', 'NIE')),
  Nr_pracownika Integer NOT NULL
)
/

-- Add keys for table Opiekuni

ALTER TABLE Opiekuni ADD CONSTRAINT Unique_Identifier10
PRIMARY KEY (Nr_pracownika)
/

-- Table Pracownicy_Zwierzęta

CREATE TABLE Pracownicy_Zwierzęta(
  Nr_pracownika Integer NOT NULL,
  Nr_zwierzecia Integer NOT NULL
)
/

-- Table Zwierzęta_Wolontariusze

CREATE TABLE Zwierzęta_Wolontariusze(
  Nr_zwierzecia Integer NOT NULL,
  Nr_wolontariusza Integer NOT NULL
)
/

-- Table Zwierzęta_Zaopatrzenie

CREATE TABLE Zwierzęta_Zaopatrzenie(
  Nr_zwierzecia Integer NOT NULL,
  Nr_egzemplarza Integer NOT NULL
)
/

-- Table Adresy

CREATE TABLE Adresy(
  Nr_adresu Integer NOT NULL,
  Miasto Varchar2(30 ) NOT NULL,
  Ulica Varchar2(40 ) NOT NULL,
  Nr_domu Number(3,0) NOT NULL,
  Nr_mieszkania Number(3,0),
  Kod_pocztowy Char(6 ) NOT NULL
)
/

-- Add keys for table Adresy

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

-- Table Zaopatrzenie
```

```
CREATE TABLE Zaopatrzenie(
  Nr_egzemplarza Integer NOT NULL,
  Rodzaj Varchar2(7 ) NOT NULL
        CHECK (Rodzaj IN ('KARMA', 'ZABAWKA', 'SMYCZ','MISKA')),
  Nazwa Varchar2(20 ) NOT NULL,
  Dostepnosc Varchar2(3 ) NOT NULL
        CHECK (Dostepnosc IN ('TAK', 'NIE')),
  Nr_wolontariusza Integer,
  Nr_pracownika Integer,
  Nr_schroniska Integer NOT NULL
)
/

-- Create indexes for table Zaopatrzenie

CREATE INDEX IX_Relationship12 ON Zaopatrzenie (Nr_pracownika)
/

CREATE INDEX IX_Relationship17 ON Zaopatrzenie (Nr_schroniska)
/

-- Add keys for table Zaopatrzenie

ALTER TABLE Zaopatrzenie ADD CONSTRAINT PK_Zaopatrzenie
PRIMARY KEY (Nr_egzemplarza)
/

-- Table Pensje

CREATE TABLE Pensje(
  Nr_pensji Integer NOT NULL,
  Kwota Float(2) NOT NULL,
  Data Date NOT NULL,
  Nr_pracownika Integer
)
/

-- Create indexes for table Pensje

CREATE INDEX IX_Relationship7 ON Pensje (Nr_pracownika)
/

-- Add keys for table Pensje

ALTER TABLE Pensje ADD CONSTRAINT PK_Pensje
PRIMARY KEY (Nr_pensji)
/

-- Table Zalozyciele

CREATE TABLE Zalozyciele(
  Nr_zalozyciela Integer NOT NULL,
  Imie Char(20 ) NOT NULL,
  Naziwska Varchar2(30 ) NOT NULL,
```

```
  Nr_schroniska Integer NOT NULL
)
/

-- Create indexes for table Zalozyciele

CREATE INDEX IX_Relationship8 ON Zalozyciele (Nr_schroniska)
/

-- Add keys for table Zalozyciele

ALTER TABLE Zalozyciele ADD CONSTRAINT PK_Zalozyciele
PRIMARY KEY (Nr_zalozyciela)
/

-- Table Obowiazki

CREATE TABLE Obowiazki(
  Nr_obowiazku Integer NOT NULL,
  Nazwa Char(20 ) NOT NULL
)
/

-- Add keys for table Obowiazki

ALTER TABLE Obowiazki ADD CONSTRAINT PK_Obowiazki
PRIMARY KEY (Nr_obowiazku)
/

-- Table Opiekuni_obowiazki

CREATE TABLE Opiekuni_obowiazki(
  Nr_pracownika Integer NOT NULL,
  Nr_obowiazku Integer NOT NULL
)
/

-- Add keys for table Opiekuni_obowiazki

ALTER TABLE Opiekuni_obowiazki ADD CONSTRAINT PK_Opiekuni_obowiazki
PRIMARY KEY (Nr_pracownika,Nr_obowiazku)
/

-- Table Schroniska_Adoptujacy

CREATE TABLE Schroniska_Adoptujacy(
  Nr_adoptujacego Integer NOT NULL,
  Nr_schroniska Integer NOT NULL
)
/

-- Add keys for table Schroniska_Adoptujacy

ALTER TABLE Schroniska_Adoptujacy ADD CONSTRAINT PK_Schroniska_Adoptujacy
```

```
PRIMARY KEY (Nr_adoptujacego,Nr_schroniska)
/

-- Table Rodzaj_zwierzecia

CREATE TABLE Rodzaj_zwierzecia(
  Nr_rodzaju Integer NOT NULL,
  Gatunek Varchar2(4 ) NOT NULL
        CHECK (Gatunek IN ('PIES', 'KOT', 'INNE')),
  Rasa Varchar2(20 ) NOT NULL
)
/

-- Add keys for table Rodzaj_zwierzecia

ALTER TABLE Rodzaj_zwierzecia ADD CONSTRAINT PK_Rodzaj_zwierzecia PRIMARY KEY (Nr_rodzaju)
/

-- Trigger for sequence Schroniska_seq1 for column Nr_schroniska in table Schroniska ---------
CREATE OR REPLACE TRIGGER ts_Schroniska_Schroniska_seq1 BEFORE INSERT
ON Schroniska FOR EACH ROW
BEGIN
  :new.Nr_schroniska := Schroniska_seq1.nextval;
END;
/
CREATE OR REPLACE TRIGGER tsu_Schroniska_Schroniska_seq1 AFTER UPDATE OF Nr_schroniska
ON Schroniska FOR EACH ROW
BEGIN
  RAISE_APPLICATION_ERROR(-20010,'Cannot update column
  Nr_schroniska in table Schroniska as it uses sequence.');
END;
/

-- Trigger for sequence Pracownicy_seq1 for column Nr_pracownika in table Pracownicy ---------
CREATE OR REPLACE TRIGGER ts_Pracownicy_Pracownicy_seq1 BEFORE INSERT
ON Pracownicy FOR EACH ROW
BEGIN
  :new.Nr_pracownika := Pracownicy_seq1.nextval;
END;
/
CREATE OR REPLACE TRIGGER tsu_Pracownicy_Pracownicy_seq1 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 Zwierzeta_seq1 for column Nr_zwierzecia in table Zwierzeta ---------
CREATE OR REPLACE TRIGGER ts_Zwierzeta_Zwierzeta_seq1 BEFORE INSERT
ON Zwierzeta FOR EACH ROW
BEGIN
  :new.Nr_zwierzecia := Zwierzeta_seq1.nextval;
END;
```

```
/
CREATE OR REPLACE TRIGGER tsu_Zwierzeta_Zwierzeta_seq1 AFTER UPDATE OF Nr_zwierzecia
ON Zwierzeta FOR EACH ROW
BEGIN
  RAISE_APPLICATION_ERROR(-20010,'Cannot update column
  Nr_zwierzecia in table Zwierzeta as it uses sequence.');
END;
/

-- Trigger for sequence Adoptujacy_seq1 for column Nr_adoptujacego in table Adoptujacy ---------
CREATE OR REPLACE TRIGGER ts_Adoptujacy_Adoptujacy_seq1 BEFORE INSERT
ON Adoptujacy FOR EACH ROW
BEGIN
  :new.Nr_adoptujacego := Adoptujacy_seq1.nextval;
END;
/
CREATE OR REPLACE TRIGGER tsu_Adoptujacy_Adoptujacy_seq1 AFTER UPDATE OF Nr_adoptujacego
ON Adoptujacy FOR EACH ROW
BEGIN
  RAISE_APPLICATION_ERROR(-20010,'Cannot update column
  Nr_adoptujacego in table Adoptujacy as it uses sequence.');
END;
/

-- Trigger for sequence Wolontariusze_seq1 for column Nr_wolontariusza in table Wolontariusze --------
CREATE OR REPLACE TRIGGER ts_Wolontariusze_Wolontarius_0 BEFORE INSERT
ON Wolontariusze FOR EACH ROW
BEGIN
  :new.Nr_wolontariusza := Wolontariusze_seq1.nextval;
END;
/
CREATE OR REPLACE TRIGGER tsu_Wolontariusze_Wolontariu_0 AFTER UPDATE OF Nr_wolontariusza
ON Wolontariusze FOR EACH ROW
BEGIN
  RAISE_APPLICATION_ERROR(-20010,'Cannot update column
  Nr_wolontariusza in table Wolontariusze as it uses sequence.');
END;
/

-- Trigger for sequence Adres_seq1 for column Nr_adresu in table Adresy ---------
CREATE OR REPLACE TRIGGER ts_Adresy_Adres_seq1 BEFORE INSERT
ON Adresy FOR EACH ROW
BEGIN
  :new.Nr_adresu := Adres_seq1.nextval;
END;
/
CREATE OR REPLACE TRIGGER tsu_Adresy_Adres_seq1 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 Egzemplarze_seq1 for column Nr_egzemplarza in table Zaopatrzenie ---------
CREATE OR REPLACE TRIGGER ts_Zaopatrzenie_Egzemplarze__0 BEFORE INSERT
ON Zaopatrzenie FOR EACH ROW
BEGIN
  :new.Nr_egzemplarza := Egzemplarze_seq1.nextval;
END;
/
CREATE OR REPLACE TRIGGER tsu_Zaopatrzenie_Egzemplarze_0 AFTER UPDATE OF Nr_egzemplarza
ON Zaopatrzenie FOR EACH ROW
BEGIN
  RAISE_APPLICATION_ERROR(-20010,'Cannot update column
  Nr_egzemplarza in table Zaopatrzenie as it uses sequence.');
END;
/

-- Trigger for sequence Pensje_seq1 for column Nr_pensji in table Pensje ---------
CREATE OR REPLACE TRIGGER ts_Pensje_Pensje_seq1 BEFORE INSERT
ON Pensje FOR EACH ROW
BEGIN
  :new.Nr_pensji := Pensje_seq1.nextval;
END;
/
CREATE OR REPLACE TRIGGER tsu_Pensje_Pensje_seq1 AFTER UPDATE OF Nr_pensji
ON Pensje FOR EACH ROW
BEGIN
  RAISE_APPLICATION_ERROR(-20010,'Cannot update column
  Nr_pensji in table Pensje as it uses sequence.');
END;
/

-- Trigger for sequence Zalozyciel_seq1 for column Nr_zalozyciela in table Zalozyciele ---------
CREATE OR REPLACE TRIGGER ts_Zalozyciele_Zalozyciel_seq1 BEFORE INSERT
ON Zalozyciele FOR EACH ROW
BEGIN
  :new.Nr_zalozyciela := Zalozyciel_seq1.nextval;
END;
/
CREATE OR REPLACE TRIGGER tsu_Zalozyciele_Zalozyciel_s_0 AFTER UPDATE OF Nr_zalozyciela
ON Zalozyciele FOR EACH ROW
BEGIN
  RAISE_APPLICATION_ERROR(-20010,'Cannot update column
  Nr_zalozyciela in table Zalozyciele as it uses sequence.');
END;
/

-- Trigger for sequence Obowiazki_seq1 for column Nr_obowiazku in table Obowiazki ---------
CREATE OR REPLACE TRIGGER ts_Obowiazki_Obowiazki_seq1 BEFORE INSERT
ON Obowiazki FOR EACH ROW
BEGIN
  :new.Nr_obowiazku := Obowiazki_seq1.nextval;
END;
/
CREATE OR REPLACE TRIGGER tsu_Obowiazki_Obowiazki_seq1 AFTER UPDATE OF Nr_obowiazku
ON Obowiazki FOR EACH ROW
```

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


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

ALTER TABLE Zwierzeta ADD CONSTRAINT jest_zamieszkiwane FOREIGN KEY (Nr_schroniska)
REFERENCES Schroniska (Nr_schroniska)
/



ALTER TABLE Wolontariusze ADD CONSTRAINT Jest_wspierane FOREIGN KEY (Nr_schroniska)
REFERENCES Schroniska (Nr_schroniska)
/



ALTER TABLE Pracownicy ADD CONSTRAINT Zatrudnia FOREIGN KEY (Nr_schroniska)
REFERENCES Schroniska (Nr_schroniska)
/



ALTER TABLE Zwierzeta ADD CONSTRAINT Jest_adoptowany FOREIGN KEY (Nr_adoptujacego)
REFERENCES Adoptujacy (Nr_adoptujacego)
/



ALTER TABLE Pracownicy ADD CONSTRAINT Pracownik_andres FOREIGN KEY (Nr_adresu)
REFERENCES Adresy (Nr_adresu)
/



ALTER TABLE Schroniska ADD CONSTRAINT Adres_schronisko FOREIGN KEY (Nr_adresu)
REFERENCES Adresy (Nr_adresu)
/



ALTER TABLE Wolontariusze ADD CONSTRAINT Wolontariusz_adres FOREIGN KEY (Nr_adresu)
REFERENCES Adresy (Nr_adresu)
/



ALTER TABLE Adoptujacy ADD CONSTRAINT Adoptujacy_adres FOREIGN KEY (Nr_adresu)
REFERENCES Adresy (Nr_adresu)
/
```

```
ALTER TABLE Pensje ADD CONSTRAINT Pracownik_pensja FOREIGN KEY (Nr_pracownika)
REFERENCES Pracownicy (Nr_pracownika)
/


ALTER TABLE Zalozyciele ADD CONSTRAINT Schronisko_zalozyciel FOREIGN KEY (Nr_schroniska)
REFERENCES Schroniska (Nr_schroniska)
/


ALTER TABLE Opiekuni_obowiazki ADD CONSTRAINT ma_obowizek FOREIGN KEY (Nr_pracownika)
REFERENCES Opiekuni (Nr_pracownika)
/


ALTER TABLE Opiekuni_obowiazki ADD CONSTRAINT Relationship10 FOREIGN KEY (Nr_obowiazku)
REFERENCES Obowiazki (Nr_obowiazku)
/


ALTER TABLE Zaopatrzenie ADD CONSTRAINT Wolontariusz_przekazuje FOREIGN KEY (Nr_wolontariusza)
REFERENCES Wolontariusze (Nr_wolontariusza)
/


ALTER TABLE Zaopatrzenie ADD CONSTRAINT Pracownik_pozyskuje FOREIGN KEY (Nr_pracownika)
REFERENCES Pracownicy (Nr_pracownika)
/


ALTER TABLE Zwierzęta_Zaopatrzenie ADD CONSTRAINT Uzytkuje_zaopatrzenie FOREIGN KEY (Nr_egzemplarza)
REFERENCES Zaopatrzenie (Nr_egzemplarza)
/


ALTER TABLE Zaopatrzenie ADD CONSTRAINT Posiada FOREIGN KEY (Nr_schroniska)
REFERENCES Schroniska (Nr_schroniska)
/


ALTER TABLE Schroniska_Adoptujacy ADD CONSTRAINT Rejestruje_Adopotujacy FOREIGN KEY (Nr_adoptujacego)
REFERENCES Adoptujacy (Nr_adoptujacego)
/
```

```
ALTER TABLE Schroniska_Adoptujacy ADD CONSTRAINT Rejestruje_Schroniska FOREIGN KEY (Nr_schroniska)
REFERENCES Schroniska (Nr_schroniska)
/


ALTER TABLE Zwierzeta ADD CONSTRAINT jest FOREIGN KEY (Nr_rodzaju)
REFERENCES Rodzaj_zwierzecia (Nr_rodzaju)
/
```

5.4 Przykłady zapytań i poleceń SQL odnoszących się do bazy danych

5.4.1 Adopcja

W tej transakcji przychodzi nowa osoba do schroniska i rejetruje się, żeby zaadoptować zwierzę. Przed transakcją wiersz dotyczący tego zwierzęcia wyglądał następująco:

	NR_ZWIERZ...	GATUNEK	RASA	IMIE	ROK_URODZENIA	NR_SCHRONISKA	NR_ADOPTUJACEGO
1	1	PIES	Maltańczyk	Pimpek	2015	4	(null)

Żeby dokonać tej transakcji, wykonaliśmy następujące instrukcje:

```
Insert into ADOPTUJACY (imie, nazwisko, pesel, nr_adresu)
values('Zofia','Kowalska','94010176583',1);

UPDATE zwierzeta
SET nr_adoptujacego = 1
WHERE nr_zwierzecia=1;
insert into Schroniska_adoptujacy(nr_adoptujacego, nr_schroniska) values (1,4);

commit;
```

Po zakończeniu tej transakcji ten sam wiersz wyglądał jak poniżej:

	NR_ZWIERZECIA	GATUNEK	RASA	IMIE	ROK_URODZENIA	NR_SCHRONISKA	NR_ADOPTUJACEGO
1	1	PIES	Maltańczyk	Pimpek	2015	4	1

Natomiast w relacji Adoptujacy pojawiła się nowa krotka z danymi nowego adoptującego.

5.4.2 Założenie nowego schroniska

:

Założenie nowego schroniska przez przykładowe dwie osoby można wykonać za pomocą poleceń:

```
  insert into schroniska (nazwa, data_zalozenia, nr_adresu) values ('ŁAPA', '2010-03-01',1);

insert into zalozyciele (imie, naziwska, nr_schroniska) values('Jan', 'Dobrowolski', 4);
insert into zalozyciele (imie, naziwska, nr_schroniska) values('Elżebieta', 'Malecka', 4);

commit;
```

Żeby wyszukać wszystkich założycieli danego schroniska należy wpisać komendę:

```
  select * from zalozyciele
where nr_schroniska=4;
```

Wyniki wyglądają następująco:

	NR_ZALOZYCIELA	IMIE	NAZIWSKA	NR_SCHRONISKA
1	1	Jan	Dobrowolski	4
2	2	Jan	Dobrowolski	4