Projekt bazy danych “Dom Seniora”

Bazy danych i Big Data

Prowadzący: dr inż. Marcin Kowalczyk

Adrian Lis 331175
Daria Shevchenko 335901

Wydział Elektroniki i Technik Informacyjnych
Politechnika Warszawska
Warszawa, 27 lutego 2025

Spis treści

1 Zakres i cel projektu (opis założeń funkcjonalnych projektowanej bazy danych)

Projekt "Dom Seniora" polegał na zaprojektowaniu i implementacji relacyjnej bazy danych, wspierającej zarządzanie placówkami opieki dla seniorów. Głównym celem była nauka projektowania baz danych na przykładzie systemu przechowywania i zarządzania informacjami dotyczącymi domów seniora, ich mieszkańców, personelu oraz powiązanych danych. Baza danych w przypadku wdrożenia, mogłaby być wykorzystywana do wspomagania codziennej działalności, poprawy organizacji pracy oraz zapewnienia wysokiego poziomu opieki nad seniorami.

2 Definicja systemu

	Opis
Zarządzanie informacjami o domach seniora	• Przechowywanie szczegółowych danych o placówkach, takich jak: – nazwa, data założenia, liczba pokoi, maksymalna liczba mieszkańców, – opis placówki, średnia cena zakwaterowania, informacja o pracy całodobowej, – lokalizacja placówki (adres) oraz powiązani sponsorzy.
Rejestracja i obsługa mieszkańców (seniorów)	• Gromadzenie danych osobowych, takich jak: – imię, nazwisko, PESEL, płeć, data urodzenia, – informacje o przyjęciu i wypisie z placówki. • Powiązanie seniorów z przypisanymi pokojami. • Przechowywanie informacji o stanie zdrowia mieszkańców za pomocą szczegółowych kart zdrowia.
Obsługa krewnych mieszkańców	• Rejestrowanie danych kontaktowych krewnych, ich relacji z mieszkańcami oraz zgody na kontakt.
Zarządzanie personelem	• Rejestracja pracowników, w tym: – dane personalne, daty zatrudnienia i urodzenia, – powiązanie z miejscem pracy (domem seniora), stanowiskami oraz wynagrodzeniami. • Wsparcie relacji między pracownikami a seniorami, np. przydział opieki.
Zarządzanie pokojami	• Przechowywanie informacji o pokojach, takich jak: – numer pokoju, piętro, liczba łóżek, status (np. wolny/zajęty), cena za dzień, – powiązanie pokoi z odpowiednim domamem seniora. • Rejestracja urządzeń bezpieczeństwa (czujników) zamontowanych w pokojach.
Wsparcie operacyjne i administracyjne	• Ewidencja adresów, wspólnych dla różnych encji, takich jak domy seniora, seniorzy, pracownicy czy krewni. • Rejestracja stanowisk pracowników, z opisem ich obowiązków i wymagań. • Obsługa sponsorów wspierających domy seniora oraz rejestracja otrzymywanych środków finansowych.

Rola	Cele i Funkcje systemu
Administratorzy Domów Seniora	**Cele:** Zarządzanie placówką, w tym informacjami o dostępnych pokojach, liczbie mieszkańców oraz pracownikach. Śledzenie stanu finansowego placówki (np. opłaty od mieszkańców, wsparcie sponsorów, wynagrodzenia personelu). **Funkcje systemu:** Dostęp do informacji o zajętości pokoi i aktualnej liczbie mieszkańców. Możliwość generowania raportów finansowych. Zarządzanie relacjami z sponsorami i rejestracja ich wsparcia finansowego.
Pracownicy Administracyjni	**Cele:** Obsługa dokumentacji mieszkańców i pracowników. Utrzymywanie aktualnych danych w systemie. **Funkcje systemu:** Dodawanie i aktualizacja danych osobowych mieszkańców, pracowników oraz ich krewnych. Obsługa harmonogramów pracy oraz danych o stanowiskach pracowników.
Personel Medyczny i Opiekuńczy	**Cele:** Monitorowanie stanu zdrowia mieszkańców. Usprawnienie opieki poprzez dostęp do aktualnych danych. **Funkcje systemu:** Przeglądanie i aktualizacja kart zdrowia mieszkańców. Rejestracja wyników badań (np. pomiarów ciśnienia, poziomu cukru i cholesterolu). Planowanie i przypisywanie zadań opiekuńczych w zależności od indywidualnych potrzeb seniorów.
Krewni Seniorów	**Cele:** Utrzymanie kontaktu z placówką i bliskimi. Dostęp do podstawowych informacji o stanie zdrowia oraz samopoczuciu mieszkańców. **Funkcje systemu:** Zdalny dostęp do danych (np. po uzyskaniu zgody na kontakt). Możliwość przekazywania uwag lub próśb dotyczących opieki nad seniorem.
Seniorzy (Mieszkańcy Domów Seniora)	**Cele:** Wsparcie w codziennym funkcjonowaniu oraz bezpieczeństwo. Usprawnienie komunikacji z personelem i rodziną. **Funkcje systemu:** Możliwość zgłaszania problemów (np. awarie w pokojach, potrzeba wsparcia). Bezpieczeństwo zapewnione przez monitorowanie czujników w pokojach.

3 Model konceptualny

3.1 Definicja zbiorów encji określonych w projekcie (decyzje projektowe)

Encja 'Dom_Seniora'

PUI	Nazwa Atrybutu	Typ Danych	M	Opis
TAK	Id_Domu_Seniora	SmallInt	TAK	Unikalny identyfikator domu seniora
NIE	Nazwa	Varchar2(20)	TAK	Nazwa domu seniora
NIE	Adres	Varchar2(400)	TAK	Adres znajdowania domu seniora
NIE	Data_zalozenia	Date	TAK	Data otwarcia placówki
NIE	Telefon	Varchar2(15)	NIE	Numer telefonu kontaktowego
NIE	Email	Varchar2(100)	NIE	Email kontaktowy placówki
NIE	Liczba_Pokoi	Integer	TAK	Liczba pokoi dostępnych dla seniorów
NIE	Max_Liczba_Mieszkancow	Integer	TAK	Maksymalna liczba mieszkańców
NIE	Opis_Placowki	Varchar2(400)	NIE	Dodatkowy opis placówki
NIE	Czynny_Calodobowo	Boolean	TAK	Informacja, czy placówka działa 24/7
NIE	Srednia_Cena_Mieszkania	Money	TAK	Średni koszt miesięczny za pobyt
NIE	Nazwa_Sponsora	Varchar2(20)	NIE	Określa nazwę sponsora

Opis: Reprezentuje placówkę, która oferuje usługi dla osób starszych. Przechowuje dane identyfikujące dom seniora, takie jak nazwa, adres, data założenia oraz liczba pokoi i maksymalna liczba mieszkańców. Dodatkowe atrybuty, jak średnia cena pobytu czy informacje o sponsorach, pozwalają na opisanie charakterystyki domu seniora.

Encja 'Pracownik'

PUI	Nazwa Atrybutu	Typ Danych	M	Opis
TAK	Id_Pracownika	Integer	TAK	Unikalny identyfikator pracownika
NIE	Imie	Varchar2(20)	TAK	Imię pracownika
NIE	Nazwisko	Varchar2(30)	TAK	Nazwisko pracownika
NIE	Stanowisko	Varchar2(50)	TAK	Stanowisko, np. "pielęgniarz", "lekarz"
NIE	Data_Zatrudnienia	Date	TAK	Data rozpoczęcia pracy
NIE	Wynagrodzenie	Money	TAK	Miesięczne wynagrodzenie
NIE	Adres	Varchar2(400)	TAK	Adres zamieszkania pracownika
NIE	Telefon	Varchar2(15)	TAK	Telefon kontaktowy pracownika
NIE	Email	Varchar2(50)	TAK	Email pracownika
NIE	Data_Urodzenia	Date	TAK	Data urodzenia pracownika

*Opis:*Przechowuje dane o pracownikach zatrudnionych w domu seniora. Zawiera podstawowe informacje personalne, takie jak imię, nazwisko, adres oraz dane kontaktowe. Dodatkowo zapisuje stanowisko, datę zatrudnienia i wynagrodzenie, co pozwala na zarządzanie personelem placówki.

Encja 'Senior'

PUI	Nazwa Atrybutu	Typ Danych	M	Opis
TAK	Id_Seniora	Integer	TAK	Unikalny identyfikator seniora
NIE	Imie	Varchar2(20)	TAK	Imię seniora
NIE	Nazwisko	Varchar2(30)	TAK	Nazwisko seniora
NIE	DataUrodzenia	Date	TAK	Data urodzenia seniora
NIE	PESEL	Character(11)	NIE	PESEL seniora
NIE	Plec	Character(1)	TAK	Płeć seniora
NIE	Adres	Varchar2(400)	TAK	Adres zamieszkania seniora przed przyjęciem do placówki

PUI	Nazwa Atrybutu	Typ Danych	M	Opis
NIE	Telefon	Varchar2(15)	NIE	Telefon kontaktowy seniora lub jego opiekuna
NIE	DataPrzyjecia	Date	TAK	Data przyjęcia do domu seniora
NIE	StanCywilny	Varchar2(15)	NIE	Np. „panna/kawaler", „żonaty-/zamężna"
NIE	DataWypisu	Date	NIE	Data wypisu (jeśli zakończono pobyt)

Opis: Zawiera dane o mieszkańcach domu seniora. Obejmuje informacje osobiste, takie jak imię, nazwisko, data urodzenia oraz PESEL. Przechowuje również datę przyjęcia i ewentualną datę wypisu, co ułatwia śledzenie pobytu seniora w placówce. Dodatkowe dane, jak telefon do opiekuna, stan cywilny, czy płeć, pomagają w lepszym zarządzaniu relacjami z mieszkańcami.

Encja 'Karta_zdrowia'

PUI	Nazwa Atrybutu	Typ Danych	M	Opis
TAK	Id_Karty_zdrowia	Integer	TAK	Unikalny identyfikator karty zdrowia
NIE	Stan_Zdrowia	Varchar2(400)	TAK	Ogólny stan zdrowia seniora
NIE	Data_Pomiaru	Date	TAK	Data ostatniego pomiaru zdrowia
NIE	Waga	Decimal(5, 2)	TAK	Masa ciała seniora
NIE	Wzrost	SmallInt	TAK	Wzrost seniora
NIE	Cisnienie_gorne	Integer	TAK	Ciśnienie górne
NIE	Cisnienie_dolne	Integer	NIE	Ciśnienie dolne
NIE	Poziom_Cukru	Decimal(5, 2)	TAK	Poziom cukru we krwi
NIE	Poziom_Cholesterolu	Decimal(5, 2)	TAK	Poziom cholesterolu

Opis: Reprezentuje stan zdrowia mieszkańców. Przechowuje szczegółowe informacje medyczne, takie jak wzrost, waga, poziom cukru i cholesterolu, a także ciśnienie krwi. Dane te są regularnie aktualizowane, co umożliwia monitorowanie stanu zdrowia mieszkańców.

Encja 'Krewny'

PUI	Nazwa Atrybutu	Typ Danych	M	Opis
TAK	Id_Krewnego	Integer	TAK	Unikalny identyfikator krewnego
NIE	Imie	Varchar2(20)	TAK	Imię członka rodziny
NIE	Nazwisko	Varchar2(30)	TAK	Nazwisko członka rodziny
NIE	Relacja	Varchar2(30)	TAK	Np. „syn", „córka"
NIE	Telefon	Varchar2(15)	TAK	Telefon kontaktowy
NIE	email	Varchar2(50)	TAK	Email do kontaktu
NIE	Adres	Varchar2(400)	TAK	Adres zamieszkania członka rodziny
NIE	Zgoda_Na_Kontakt	Boolean	TAK	Zgoda na kontakt w sprawach seniora

Opis: Zawiera dane kontaktowe najbliższych członków rodziny seniora. Uwzględnia takie informacje jak imię, nazwisko, telefon, email oraz relację do seniora (np. syn, córka). Atrybut zgoda na kontakt pozwala sprawdzić, czy krewny wyraził zgodę na kontakt w sprawach seniora.

Encja 'Pokoj'

PUI	Nazwa Atrybutu	Typ Danych	M	Opis
TAK	Id_Pokoju	Integer	TAK	Unikalny identyfikator pokoju
NIE	Numer_Pokoju	Varchar2(10)	TAK	Numer pokoju
NIE	Pietro	Integer	TAK	Piętro, na którym znajduje się pokój

PUI	Nazwa Atrybutu	Typ Danych	M	Opis
NIE	Liczba_Lozek	Integer	TAK	Liczba łóżek w pokoju
NIE	Status_Pokoju	Varchar2(20)	TAK	Status pokoju (np. "dostępny", "zajęty")
NIE	Cena_Za_Dzien	Money	TAK	Cena za dzień pobytu w pokoju
NIE	Opis	Varchar2(400)	NIE	Dodatkowy opis pokoju (np. widok, wyposażenie)

Opis: Zawiera informacje dotyczące pokoi w domu seniora uwzględniające jego parametry techniczne i stan aktualny.

Encja 'Czujnik_Bezpieczeństwa'

PUI	Nazwa Atrybutu	Typ Danych	M	Opis
TAK	Id_czujnika	Integer	TAK	Unikalny identyfikator czujnika
NIE	Typ	Varchar2(20)	TAK	Określa przed czym ostrzega dany czujnik
NIE	Ostatnia_Aktywacja	DateTime	NIE	Określa, kiedy czujnik ostatnio aktywował alarm

Opis: Służy do monitorowania bezpieczeństwa w domu seniora. Rejestruje typ czujnika (np. dym, ruch) oraz ostatnią aktywację. Ułatwia nadzór nad systemem alarmowym i dbałość o bezpieczeństwo mieszkańców.

3.2 Ustalenie związków między encjami i ich typów

Nazwa Relacji	Relacja Między	Moc Relacji
Gosci	Dom_Seniora - Senior	1..1 - 0..m

Opis: Każdy Dom Seniora może gościć wielu seniorów (0..m), ale każdy senior jest przypisany do jednego domu seniora (1..1).

Zatrudnia	Dom_Seniora - Pracownik	1..1 - 0..m

Opis: Każdy Dom Seniora może zatrudniać wielu pracowników (0..m), ale każdy pracownik jest przypisany do jednego domu seniora (1..1).

Opiekuje_sie	Pracownik - Senior	0..n - 0..m

Opis: Jeden pracownik może opiekować się wieloma seniorami (0..n), a każdy senior może być pod opieką wielu pracowników (0..m).

Jest_Opisany_Przez	Senior - Karta_zdrowia	1..1 - 0..1

Opis: Każdy senior może mieć maksymalnie jedną kartę zdrowia (0..1), a każda karta zdrowia należy do jednego seniora (1..1).

Jest_Powiazany	Senior - Krewny	1..1 - 0..m

Opis: Każdy senior może być powiązany z wieloma krewnymi (0..m), ale każdy krewny należy do jednego seniora (1..1).

Zawiera	Dom_Seniora - Pokoj	1..1 - 0..m

Opis: Każdy Dom Seniora może zawierać wiele pokoi (0..m), ale każdy pokój należy do jednego konkretnego domu seniora (1..1).

Zajmuje	Senior - Pokoj	0..n - 0..1

Opis: Senior może zajmować jeden pokój (0..1), ale pokój może być pusty lub przeznaczony dla wielu mieszkańców (0..n).

Ma_zamontowany	Pokoj - Czujnik_Bezpieczenstwa	1..1 - 0..m

Opis: Każdy pokój może mieć zamontowane wiele czujników bezpieczeństwa (0..m), ale każdy czujnik należy do jednego konkretnego pokoju (1..1).

3.3 Dodatkowe reguły integralnościowe (reguły biznesowe)

Domena	**Zasada**	**Typ danych**
Plec	Plec IN ('M', 'K')	Char(1)
Relacja	Relacja IN ('syn', 'córka', 'żona', 'mąż', 'inny')	Varchar2(12)

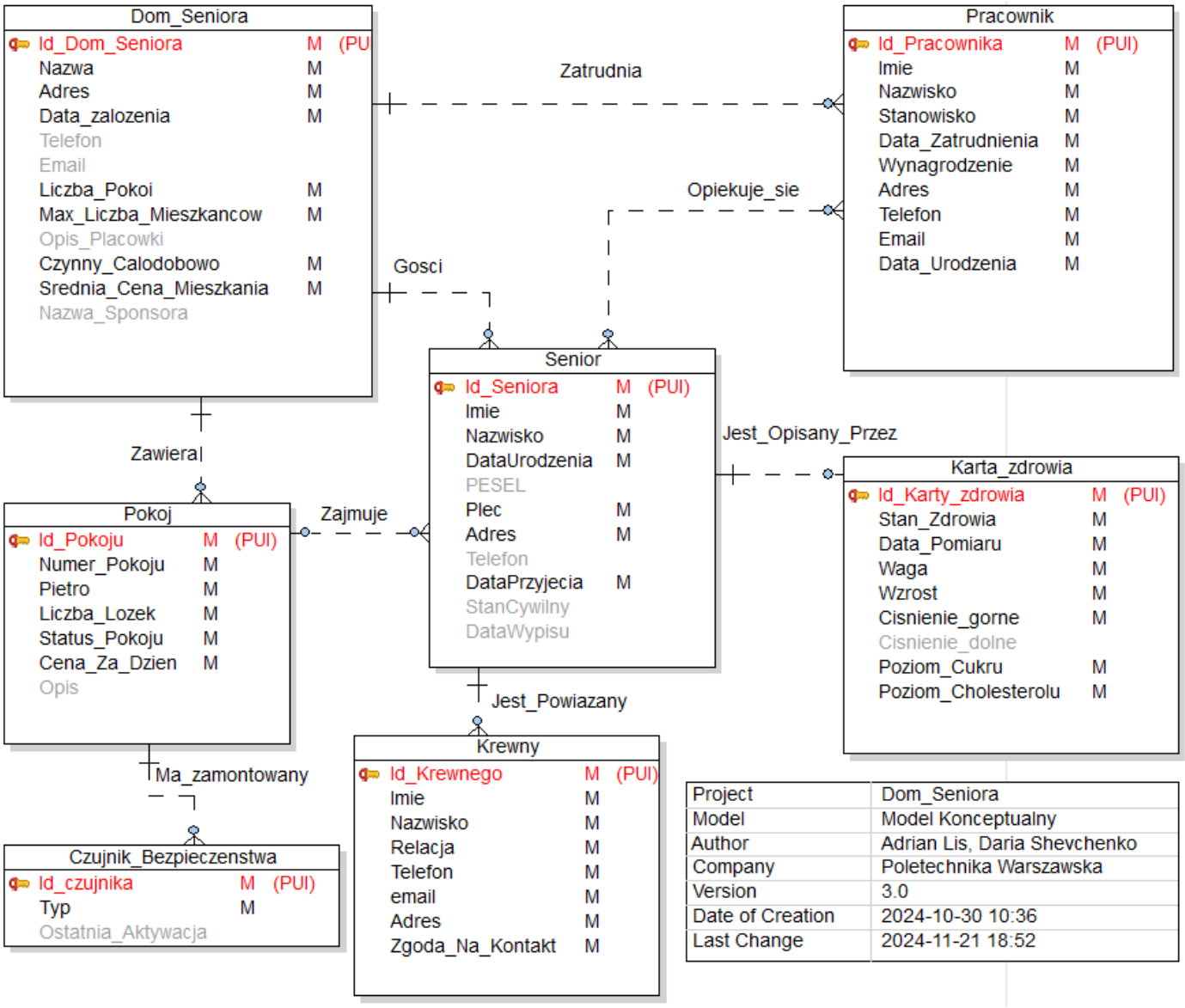
Opis: W bazie danych zdefinowaliśmy domenę Plec, która reprezentuje płeć seniora w systemie, z ograniczeniem wartości do M (mężczyzna) lub K (kobieta), oraz z typem danych określonym jako pojedynczy znak (Char(1)).
Druga domena, 'Relacja', określa rodzaj relacji seniora do osoby wpisanej w systemie, z możliwymi wartościami: syn, córka, żona, mąż, oraz inny. Typ danych tej domeny to Varchar2(12), co pozwala na zapisanie wartości o zmiennym rozmiarze do 12 znaków.

3.4 Klucze kandydujące i główne (decyzje projektowe)

Encja	**Klucz główny**	**Klucze kandydujące**	**Wyjaśnienie**
Dom_Seniora	Id_Domu_Seniora	Nazwa, Adres	Klucze kandydujące to Nazwa i Adres, ponieważ kombinacja tych atrybutów może jednoznacznie identyfikować dom seniora w systemie.
Pracownik	Id_Pracownika	Imie, Nazwisko	Imie i Nazwisko mogą być użyte jako klucze kandydujące, jednak nie zawsze są jednoznaczne.
Senior	Id_Seniora	PESEL, Imie, Nazwisko	Klucze kandydujące to PESEL (unikalny numer identyfikacyjny) oraz kombinacja Imie i Nazwisko, które mogą również służyć do identyfikacji, jednak nie zawsze będą jednoznaczne.
Karta Zdrowia	Id_Karty_zdrowia	Brak	Brak kluczy kandydujących, ponieważ nie istnieje inny zestaw atrybutów, który mógłby jednoznacznie identyfikować rekord karty zdrowia.
Krewny	Id_Krewnego	Imie, Nazwisko, Relacja	Klucze kandydujące to kombinacja Imie, Nazwisko, Relacja, które mogą jednoznacznie identyfikować członka rodziny, ale w praktyce mogą występować powtórzenia.
Pokoj	Id_Pokoju	Numer_Pokoju	Klucz kandydujący to Numer_Pokoju, ponieważ może on jednoznacznie identyfikować pokój w placówce.
Czujnik_Bezpieczeństwa	Id_czujnika	Typ, Ostatnia_Aktywacja	Klucze kandydujące to Typ i Ostatnia_Aktywacja, które mogą pomóc w identyfikacji czujnika, zwłaszcza w przypadku wielu czujników tego samego typu.

Opis: Tabela zawiera klucze główne oraz kandydujące, które są przypisane do encji systemu. Klucz główny jest unikalnym identyfikatorem rekordu, a klucze kandydujące to atrybuty, które również mogą służyć do identyfikacji, ale mogą nie być w pełni jednoznaczne. Wyjaśnienia wskazują, które atrybuty są używane do tego celu i w jakim kontekście mogą lub nie mogą zapewnić unikalności.

3.5 Schemat ER na poziomie konceptualnym



Rysunek 1: Schemat Konceptualny

3.6 Problem pułapek szczelinowych i wachlarzowych – analiza i przykłady

Projektując bazę danych natknęliśmy się na pułapkę szczelinową. We wstępnej wersji projektu bazy danych Pracownik nie był w żaden sposób powiązany z Domem Seniora, a jedynie z Seniorem. Prowadziło to do sytuacji, w której pracownik który nie jest już lub jeszcze zatrudniony mógłby zajmować się seniorem. Pułapka została wykryta podczas konsultacji z koordynatorem projektu i szybko usunięta poprzez dodanie relacji między encjami Pracownik i Dom seniora.

4 Model logiczny

4.1 Charakterystyka modelu relacyjnego

Model relacyjny tworzy się, usuwając związki wiele do wielu i zastępując je dwoma związkami jeden do wielu. Encje zamienia się na tabele, nadając im nazwy w liczbie mnogiej zamiast pojedynczej. Związki jeden do wielu odwzorowuje się przez dodanie klucza obcego w tabeli po stronie „wielu", powiązanego z kluczem głównym tabeli po stronie „jeden". Model logiczny uwzględnia też charakterystykę systemu bazy danych, w tym przypadku jest to Oracle

4.2 Usunięcie właściwości niekompatybilnych z modelem relacyjnym - przykłady

4.2.1 Opis problemów związków Wiele-do-Wielu

Niekompatybilne z modelem relacyjnym są związki *wiele-do-wielu*. Relacje *wiele-do-wielu* pojawiają się, gdy jedna encja może być powiązana z wieloma rekordami innej encji, a także na odwrót. W relacyjnych bazach danych takie związki nie mogą być bezpośrednio odwzorowane za pomocą kluczy obcych, dlatego konieczne jest utworzenie tabel pośredniczących.

Przykład związków Wiele-do-Wielu w projekcie

- **Pracownicy – Seniorzy:** Jeden pracownik może opiekować się wieloma seniorami, a jeden senior może być pod opieką wielu pracowników.

Tabela Pośrednicząca: `Pracownicy_Seniorzy`

Aby zrealizować związek *wiele-do-wielu* między tabelami `Pracownicy` i `Seniorzy`, utworzono tabelę pośredniczącą `Pracownicy_Seniorzy`, która zawiera:

- **Id_Pracownik:** Klucz obcy wskazujący na tabelę `Pracownicy`.
- **Id_Seniora:** Klucz obcy wskazujący na tabelę `Seniorzy`.
- **Data:** Informacja o dacie podjęcia opieki przez pracownika nad seniorem.
- **Uwagi:** Dodatkowe informacje związane z opieką.

4.2.2 Opis problemu dotycącego wartości boolean

Na etapie modelu konceptualnego zostały zastosowane wartości boolean, które nie są bezpośrednio kompatybilne z relacyjną bazą danych Oracle. Automatyczna konwersja w programie Toad Data Modeler ustaliła typ dawnych atrybutów boolean jako char(1). Choć jest to w ogólności poprawny typ, to koniecznym okazało się dodanie odpowiednich domen dla tych atrybutów ograniczających przyjmowane przez nie wartości zapewniając spójność bazy danych.

Czynny_Calodobowo (Tabele: `Domy_Seniora`**)**

Domena	Zasada	Typ danych
Czynny_Calodobowo	CHECK (Czynny_Calodobowo IN ('T', 'N'))	Char(1)

Opis: Zdefiniowana domena określa, czy dom seniora działa w trybie całodobowym. Dopuszczalne wartości to T (tak) oraz N (nie). Typ danych to Char(1), co oznacza, że wartość tej domeny składa się z pojedynczego znaku. Zastosowane ograniczenie dodatkowe ograniczenie NOT NULL zapewnia, że informacja o trybie pracy domu seniora musi być zawsze dostępna.

Status_Pokoju (Tabele: `Pokoje`**)**

Domena	Zasada	Typ danych
Status_Pokoju	CHECK (Status_Pokoju IN ('W', 'Z', 'R'))	Char(1)

Opis: Domena określa bieżący status pokoju w domu seniora. Dopuszczalne wartości to: W (Wolny) – pokój jest dostępny, Z (Zajęty) – pokój jest aktualnie zamieszkany, oraz R (W remoncie) – pokój jest czasowo wyłączony z użytkowania. Typ danych to Char(1), co oznacza, że wartość tej domeny składa się z pojedynczego znaku. Zastosowane dodatkowe ograniczenie NOT NULL zapewnia, że stan pokoju zawsze musi być określony.

Zgoda_Na_Kontakt (Tabele: `Krewni`**)**

Domena	Zasada	Typ danych
Zgoda_Na_Kontakt	CHECK (Zgoda_Na_Kontakt IN ('T', 'N'))	Char(1)

Opis: Domena informuje, czy krewny seniora wyraził zgodę na kontakt w nagłych wypadkach lub do innych celów. Dopuszczalne wartości to: T (tak) – zgoda została wyrażona, oraz N (nie) – zgoda nie została wyrażona. Typ danych to Char(1), co oznacza, że wartość tej domeny składa się z pojedynczego znaku. Zastosowane dodatkowe ograniczenie NOT NULL zapewnia, że decyzja o zgodzie musi być zawsze zarejestrowana.

4.3 Proces normalizacji – analiza i przykłady

Cel Normalizacji

Proces normalizacji w projektowanej bazie danych miał na celu eliminację redundancji danych oraz zapewnienie spójności i integralności danych w SZBD. Normalizacja bazy danych przebiegała przez kolejne postacie normalne (1NF, 2NF, 3NF). Poniżej przedstawiono, w jaki sposób zostały spełnione wymagania każdej z tych postaci:

Proces Normalizacji

Encja 'Sponsorzy'

Klucz	Nazwa Atrybutu	Typ Danych	M	Opis
PK	Id_Sponsora	Integer	TAK	Unikalny identyfikator sponsora
	Nazwa	Varchar2(20)	TAK	Nazwa Sponsora
	Kwota_Miesiecznego_Wsparcia	Integer	TAK	Kwota wpłacana przez sponsora co miesiąc na rzecz domu seniora

Encja 'Adresy'

Klucz	Nazwa Atrybutu	Typ Danych	M	Opis
PK	Id_Adresu	Integer	TAK	Unikalny identyfikator
	Miasto	Varchar2(20)	TAK	Nazwa miasta
	Ulica	Varchar2(30)	TAK	Nazwa ulicy
	Nr_lokalu	Varchar2(5)	NIE	Numer lokalu
	Nr_budynku	Varchar2(5)	TAK	Numer budynku

Encja 'Wynagrodzenia'

Klucz	Nazwa Atrybutu	Typ Danych	M	Opis
PK	Id_Wynagrodzenia	Integer	TAK	Unikalny identyfikator wynagrodzenai
	Data	Date	TAK	Data wypłaty wynagrodzenia
	Kwota	Number(10,2)	TAK	Kwota wynagrodzenia
	Kwota_dod	Number(10,2)	NIE	Dodatkowa kwota wynagrodzenia
FK	Id_Pracownika	Integer	TAK	Unikalny identyfikator pracownika

Pierwsza Postać Normalna (1NF)

- **Brak wielowartościowych atrybutów:** Każda kolumna tabeli przechowuje dokładnie jedną wartość. Przykładowo:
  - Użyte zostały osobne atrybuty na *Imię* i *Nazwisko*. Nie me atrybutu *Imię i Nazwisko* który przechowywałby jednocześnie imię i nazwisko
- **Brak powtarzających się grup** Każda tabela przechowuje dane jednego typu, np. tabela `Domy_Seniora` przechowuje informacje o domach seniora, a `Adresy` o adresach.

Druga Postać Normalna (2NF)

- **Brak częściowych zależności funkcyjnych:** Każdy atrybut niekluczowy w tabeli zależy od klucza a nie od jego części.
  - Przykładem jest tabela `Pokoje`, w której wszystkie informacje, takie jak `Numer_Pokoju`, `Pietro`, czy `Status_Pokoju`, są w pełni zależne od klucza głównego `Id_Pokoju`.

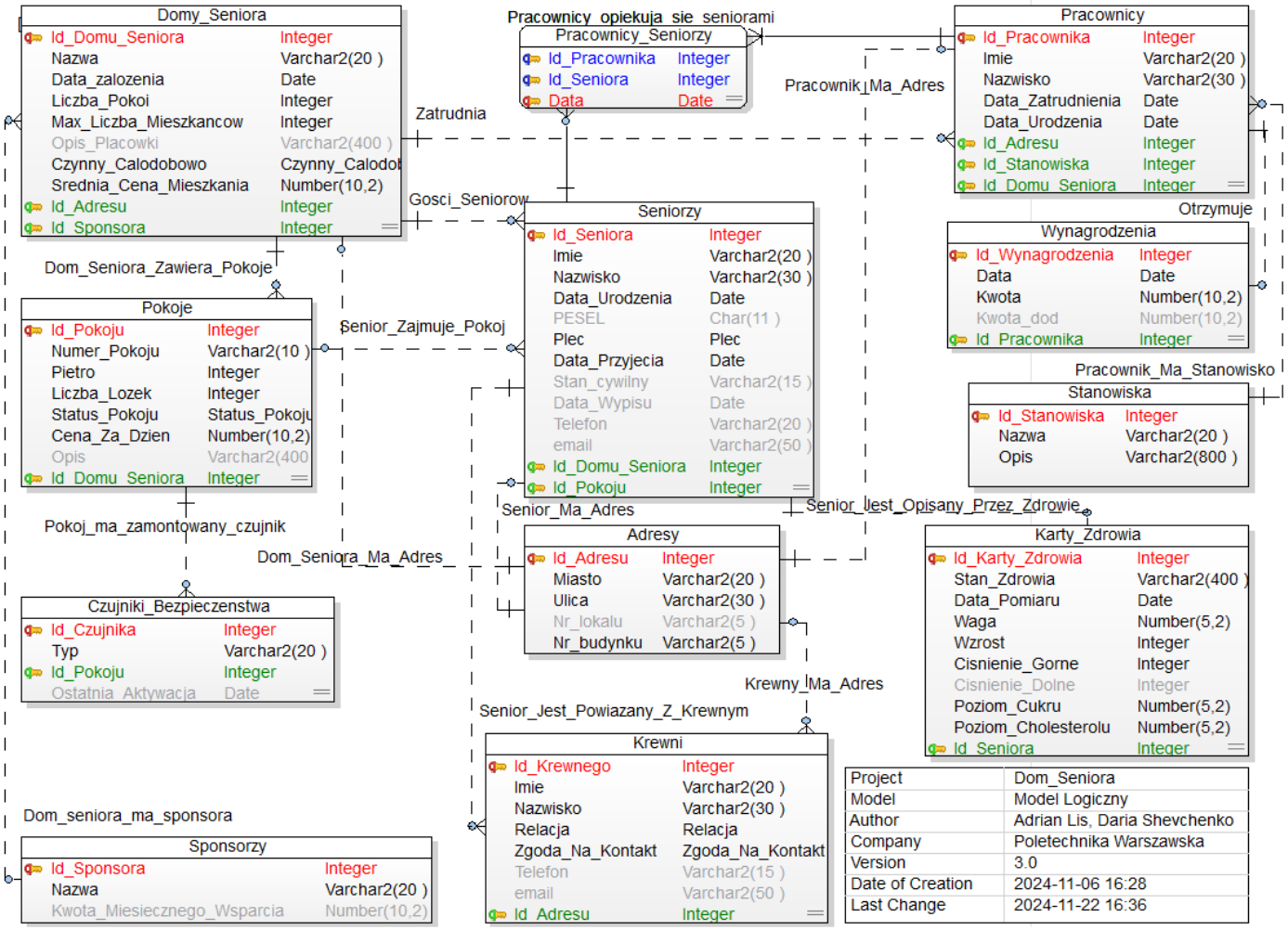
Trzecia Postać Normalna (3NF)

- **Brak zależności przechodnich:** Każdy atrybut relacji nie wchodzący w skład żadnego klucza potencjalnego nie jest przechodnio funkcyjnie zależny od żadnego klucza potencjalnego tej relacji. "Wszystkie niekluczowe kolumny są określone kluczem, całym kluczem i tylko kluczem"
  - Przykładowo W tabeli `Pracownicy`, każdy atrybut jest zależny wyłącznie od `Id_Pracownika`.
  - W tabeli `Wynagrodzenia`, każdy atrybut jest zależny wyłącznie od `Id_Wynagrodzenia`.

Ostateczna postać normalna

Po zakończeniu procesu normalizacji baza danych znajduje się w **trzeciej postaci normalnej (3NF)**. Wszystkie atrybuty w tabelach są w pełni zależne od kluczy głównych, a zależności przechodnie zostały wyeliminowane.

4.4 Schemat ER na poziomie modelu logicznego



Rysunek 2: Schemat Logiczny

4.5 Więzy integralności

W celu zapewnienia integralności i spójności danych zastosowano:

- Klucze główne (*PRIMARY KEY*) w każdej tabeli, gwarantujące jednoznaczną identyfikację rekordów.
- Klucze obce (*FOREIGN KEY*) definiujące relacje między tabelami, wraz z regułami kaskadowego usuwania lub aktualizacji.
- Ograniczenia (*CHECK CONSTRAINT*), np. dla Relacji (`Relacja IN ('syn','córka','żona','mąż','inny')`).

4.6 Proces denormalizacji – analiza i przykłady

Rozważania dotyczące denormalizacji rozpoczęliśmy od ustalenia, czy możliwe będzie wprowadzenie kontrolowanej redundancji poprzez złagodzenie zakresu stosowania reguł normalizacji i usyskanie w ten sposób korzyści wydajnościowych. Ostatecznie w projekcie proces denormalizacji został zastosowany jedynie w bardzo ograniczonym stopniu. We wstępnej wersji modelu logicznego występowała osobna tabela zawierająca dane kontaktowe dla np. Seniorów i ich Krewnych. Została ona jednak usunięta na rzecz atrybutów w tabelach *Senior* i *Krewny*, co może pomóc w przyspieszeniu działania systemu bazy danych.

5 Faza fizyczna

5.1 Projekt transakcji i weryfikacja ich wykonalności

Transakcja	Potrzebne zasoby	Czy realizowane
Dodawanie, usuwanie i modyfikacja danych dotyczących domów seniora	Domy_Seniora, Adresy	TAK
Podgląd danych dotyczących domów seniora	Domy_Seniora, Adresy	TAK
Dodawanie, usuwanie i modyfikacja danych dotyczących seniorów	Seniorzy, Pokoje, Adresy	TAK
Podgląd danych dotyczących seniorów	Seniorzy, Pokoje, Adresy	TAK
Dodawanie, usuwanie i modyfikacja danych dotyczących pokoi	Pokoje, Domy_Seniora	TAK
Podgląd dostępnych pokoi w domach seniora	Pokoje, Domy_Seniora	TAK
Dodawanie, usuwanie i modyfikacja danych dotyczących krewnych	Krewni, Adresy	TAK
Podgląd danych dotyczących krewnych	Krewni, Seniorzy, Adresy	TAK
Dodawanie, usuwanie i modyfikacja danych zdrowotnych seniorów	Karty_Zdrowia, Seniorzy	TAK
Podgląd danych zdrowotnych seniorów	Karty_Zdrowia, Seniorzy	TAK
Dodawanie, usuwanie i modyfikacja danych dotyczących pracowników	Pracownicy, Adresy, Stanowiska	TAK
Podgląd danych dotyczących pracowników	Pracownicy, Adresy, Stanowiska	TAK
Dodawanie, usuwanie i modyfikacja danych dotyczących sponsorów	Sponsorzy	TAK
Podgląd danych dotyczących sponsorów	Sponsorzy	TAK
Dodawanie, usuwanie i modyfikacja danych dotyczących czujników bezpieczeństwa	Czujniki_Bezpieczenstwa, Pokoje	TAK
Podgląd danych dotyczących czujników bezpieczeństwa	Czujniki_Bezpieczenstwa, Pokoje	TAK

5.2 Strojenie bazy danych – dobór indeksów

Table: `Domy_Seniora`

```
CREATE INDEX IX_Id_Domu_Seniora ON Domy_Seniora (Id_Domu_Seniora);
CREATE INDEX IX_Id_Adres ON Domy_Seniora (Id_Adres);
CREATE INDEX IX_Id_Sponsora ON Domy_Seniora (Id_Sponsora);
```

Table: `Seniorzy`

```
CREATE INDEX IX_Id_Seniora ON Seniorzy (Id_Seniora);
CREATE INDEX IX_Id_Dom_Seniora ON Seniorzy (Id_Dom_Seniora);
CREATE INDEX IX_Id_Pokoj ON Seniorzy (Id_Pokoj);
CREATE INDEX IX_Id_Adres ON Seniorzy (Id_Adres);
```

Table: `Pracownicy`

```
CREATE INDEX IX_Id_Pracownika ON Pracownicy (Id_Pracownika);
CREATE INDEX IX_Id_Domu_Seniora ON Pracownicy (Id_Domu_Seniora);
CREATE INDEX IX_Id_Stanowisko ON Pracownicy (Id_Stanowkisko);
CREATE INDEX IX_Id_Adres ON Pracownicy (Id_Adres);
```

Table: `Pokoje`

```
CREATE INDEX IX_Id_Pokoju ON Pokoje (Id_Pokoju);
CREATE INDEX IX_Id_Dom_Seniora ON Pokoje (Id_Dom_Seniora);
```

Table: Karty_Zdrowia

```
CREATE INDEX IX_Id_Karty_Zdrowia ON Karty_Zdrowia (Id_Karty_Zdrowia);
CREATE INDEX IX_Id_Senior ON Karty_Zdrowia (Id_Senior);
```

Table: Krewni

```
CREATE INDEX IX_Id_Krewnego ON Krewni (Id_Krewnego);
CREATE INDEX IX_Id_Seniora ON Krewni (Id_Seniora);
CREATE INDEX IX_Id_Adres ON Krewni (Id_Adres);
```

Table: Adresy

```
CREATE INDEX IX_Id_Adresu ON Adresy (Id_Adresu);
```

Table: Stanowiska

```
CREATE INDEX IX_Id_Stanowiska ON Stanowiska (Id_Stanowiska);
```

Table: Wynagrodzenia

```
CREATE INDEX IX_Id_Wynagrodzenia ON Wynagrodzenia (Id_Wynagrodzenia);
CREATE INDEX IX_Id_Pracownika ON Wynagrodzenia (Id_Pracownika);
```

Table: Czujniki_Bezpieczenstwa

```
CREATE INDEX IX_Id_Czujnika ON Czujniki_Bezpieczenstwa (Id_Czujnika);
CREATE INDEX IX_Id_Pokoju ON Czujniki_Bezpieczenstwa (Id_Pokoju);
```

Table: Sponsorzy

```
CREATE INDEX IX_Id_Sponsora ON Sponsorzy (Id_Sponsora);
```

Table: Pracownicy_Seniorzy

```
CREATE INDEX IX_Id_Pracownik ON Pracownicy_Seniorzy (Id_Pracownik);
CREATE INDEX IX_Id_Seniora ON Pracownicy_Seniorzy (Id_Seniora);
```

5.3 Skrypt SQL zakładający bazę danych

Skryp SQL został wygenerowany w Toad Data Modeler i uruchomiony na zdalnej bazie danych przy pomocy programu SQL Developer. Kod ten wykonał się poprawnie i założył bazę danych

Listing 1: Kod SQL z pliku

```
   /*
Created: 2024-11-06
Modified: 2024-11-22
Project: Dom_Seniora
Model: Model Logiczny
Company: Poletechnika Warszawska
Author: Adrian Lis, Daria Shevchenko
Version: 3.0
Database: Oracle 18c
*/

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

```
-- Table Domy_Seniora

CREATE TABLE Domy_Seniora(
   Id_Domu_Seniora Integer NOT NULL,
   Nazwa Varchar2(20 ) NOT NULL,
   Data_zalozenia Date NOT NULL,
   Liczba_Pokoi Integer NOT NULL,
   Max_Liczba_Mieszkancow Integer NOT NULL,
   Opis_Placowki Varchar2(400 ),
   Czynny_Calodobowo Char(1 ) NOT NULL
         CHECK (Czynny_Calodobowo IN ('T', 'N')),
   Srednia_Cena_Mieszkania Number(10,2) NOT NULL,
   Id_Adres Integer NOT NULL,
   Id_Sponsora Integer
)
/

-- Create indexes for table Domy_Seniora

CREATE INDEX IX_Relationship7 ON Domy_Seniora (Id_Adres)
/

CREATE INDEX IX_Relationship10 ON Domy_Seniora (Id_Sponsora)
/

-- Add keys for table Domy_Seniora

ALTER TABLE Domy_Seniora
    ADD CONSTRAINT Unique_Identifier1 PRIMARY KEY (Id_Domu_Seniora)
/

-- Table Seniorzy

CREATE TABLE Seniorzy(
   Id_Seniora 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 ('M', 'K')),
   Data_Przyjecia Date NOT NULL,
   Stan_cywilny Varchar2(15 ),
   Data_Wypisu Date,
   Telefon Varchar2(20 ),
   email Varchar2(50 ),
   Id_Dom_Seniora Integer NOT NULL,
   Id_Pokoj Integer,
   Id_Adres Integer NOT NULL
)
/

-- Create indexes for table Seniorzy

CREATE INDEX IX_Ma ON Seniorzy (Id_Dom_Seniora)
/

CREATE INDEX IX_Zajmuje ON Seniorzy (Id_Pokoj)
/

CREATE INDEX IX_Relationship12 ON Seniorzy (Id_Adres)
/

-- Add keys for table Seniorzy

ALTER TABLE Seniorzy
    ADD CONSTRAINT Unique_Identifier2 PRIMARY KEY (Id_Seniora)
/
```

```
-- Table Pracownicy

CREATE TABLE Pracownicy(
    Id_Pracownika Integer NOT NULL,
    Imie Varchar2(20 ) NOT NULL,
    Nazwisko Varchar2(30 ) NOT NULL,
    Data_Zatrudnienia Date NOT NULL,
    Data_Urodzenia Date NOT NULL,
    Id_Adres Integer NOT NULL,
    Id_Stanowkisko Integer NOT NULL,
    Id_Domu_Seniora Integer NOT NULL
)
/

-- Create indexes for table Pracownicy

CREATE INDEX IX_Relationship13 ON Pracownicy (Id_Adres)
/

CREATE INDEX IX_Relationship11 ON Pracownicy (Id_Stanowkisko)
/

CREATE INDEX IX_Relationship1 ON Pracownicy (Id_Domu_Seniora)
/

-- Add keys for table Pracownicy

ALTER TABLE Pracownicy
    ADD CONSTRAINT Unique_Identifier4 PRIMARY KEY (Id_Pracownika)
/

-- Table Karty_Zdrowia

CREATE TABLE Karty_Zdrowia(
    Id_Karty_Zdrowia Integer NOT NULL,
    Stan_Zdrowia Varchar2(400 ) NOT NULL,
    Data_Pomiaru Date NOT NULL,
    Waga Number(5,2) NOT NULL,
    Wzrost Integer NOT NULL,
    Cisnienie_Gorne Integer NOT NULL,
    Cisnienie_Dolne Integer,
    Poziom_Cukru Number(5,2) NOT NULL,
    Poziom_Cholesterolu Number(5,2) NOT NULL,
    Id_Senior Integer NOT NULL
)
/

-- Create indexes for table Karty_Zdrowia

CREATE INDEX IX_Jest_Opisany_Przez ON Karty_Zdrowia (Id_Senior)
/

-- Add keys for table Karty_Zdrowia

ALTER TABLE Karty_Zdrowia
    ADD CONSTRAINT Unique_Identifier5 PRIMARY KEY (Id_Karty_Zdrowia)
/

-- Table Krewni

CREATE TABLE Krewni(
    Id_Krewnego Integer NOT NULL,
    Imie Varchar2(20 ) NOT NULL,
    Nazwisko Varchar2(30 ) NOT NULL,
    Relacja Varchar2(12 ) NOT NULL
        CHECK (Relacja IN ('syn','c rka',' ona ','m    ','inny')),
    Zgoda_Na_Kontakt Char(1 ) NOT NULL,
```

```
  Telefon Varchar2(15 ),
  email Varchar2(50 ),
  Id_Adres Integer,
  Id_Seniora Integer NOT NULL
)
/

-- Create indexes for table Krewni

CREATE INDEX IX_Relationship6 ON Krewni (Id_Adres)
/

CREATE INDEX IX_Relationship3 ON Krewni (Id_Seniora)
/

-- Add keys for table Krewni

ALTER TABLE Krewni
    ADD CONSTRAINT Unique_Identifier6 PRIMARY KEY (Id_Krewnego)
/

-- Table Pokoje

CREATE TABLE Pokoje(
  Id_Pokoju Integer NOT NULL,
  Numer_Pokoju Varchar2(10 ) NOT NULL,
  Pietro Integer NOT NULL,
  Liczba_Lozek Integer NOT NULL,
  Status_Pokoju Char(1 ) NOT NULL,
  Cena_Za_Dzien Number(10,2) NOT NULL,
  Opis Varchar2(400 ),
  Id_Dom_Seniora Integer NOT NULL
)
/

-- Create indexes for table Pokoje

CREATE INDEX IX_Zawiera ON Pokoje (Id_Dom_Seniora)
/

-- Add keys for table Pokoje

ALTER TABLE Pokoje
    ADD CONSTRAINT Unique_Identifier7 PRIMARY KEY (Id_Pokoju)
/

-- Table Pracownicy_Seniorzy

CREATE TABLE Pracownicy_Seniorzy(
  Id_Pracownik Integer NOT NULL,
  Id_Seniora Integer NOT NULL,
  Data Date NOT NULL,
  Uwagi Varchar2(400 )
)
/

-- Table Adresy

CREATE TABLE Adresy(
  Id_Adresu Integer NOT NULL,
  Miasto Varchar2(20 ) NOT NULL,
  Ulica Varchar2(30 ) NOT NULL,
  Nr_lokalu Varchar2(5 ),
  Nr_budynku Varchar2(5 ) NOT NULL
)
/

-- Add keys for table Adresy
```

```
ALTER TABLE Adresy
    ADD CONSTRAINT PK_Adresy PRIMARY KEY (Id_Adresu)
/

-- Table Stanowiska

CREATE TABLE Stanowiska(
  Id_Stanowiska Integer NOT NULL,
  Nazwa Varchar2(20 ) NOT NULL,
  Opis Varchar2(800 ) NOT NULL
)
/

-- Add keys for table Stanowiska

ALTER TABLE Stanowiska
    ADD CONSTRAINT PK_Stanowiska PRIMARY KEY (Id_Stanowiska)
/

ALTER TABLE Stanowiska ADD CONSTRAINT Nazwa UNIQUE (Nazwa)
/

-- Table Wynagrodzenia

CREATE TABLE Wynagrodzenia(
  Id_Wynagrodzenia Integer NOT NULL,
  Data Date NOT NULL,
  Kwota Number(10,2) NOT NULL,
  Kwota_dod Number(10,2),
  Id_Pracownika Integer NOT NULL
)
/

-- Create indexes for table Wynagrodzenia

CREATE INDEX IX_Relationship2 ON Wynagrodzenia (Id_Pracownika)
/

-- Add keys for table Wynagrodzenia

ALTER TABLE Wynagrodzenia
    ADD CONSTRAINT PK_Wynagrodzenia PRIMARY KEY (Id_Wynagrodzenia)
/

-- Table Czujniki_Bezpieczenstwa

CREATE TABLE Czujniki_Bezpieczenstwa(
  Id_Czujnika Integer NOT NULL,
  Typ Varchar2(20 ) DEFAULT ON NULL 0 NOT NULL,
  Id_Pokoju Integer NOT NULL,
  Ostatnia_Aktywacja Date
)
/

-- Create indexes for table Czujniki_Bezpieczenstwa

CREATE INDEX IX_Relationship8 ON Czujniki_Bezpieczenstwa (Id_Pokoju)
/

-- Add keys for table Czujniki_Bezpieczenstwa

ALTER TABLE Czujniki_Bezpieczenstwa
    ADD CONSTRAINT PK_Czujniki_Bezpieczenstwa PRIMARY KEY (Id_Czujnika)
/

-- Table Sponsorzy
```

```
CREATE TABLE Sponsorzy(
   Id_Sponsora Integer NOT NULL,
   Nazwa Varchar2(20 ) NOT NULL,
   Kwota_Miesiecznego_Wsparcia Number(10,2) DEFAULT ON NULL 0
)
/

-- Add keys for table Sponsorzy

ALTER TABLE Sponsorzy ADD CONSTRAINT PK_Sponsorzy PRIMARY KEY (Id_Sponsora)
/


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

ALTER TABLE Pokoje
    ADD CONSTRAINT Dom_Seniora_Zawiera_Pokoje
        FOREIGN KEY (Id_Dom_Seniora) REFERENCES Domy_Seniora (Id_Domu_Seniora)
/



ALTER TABLE Seniorzy
    ADD CONSTRAINT Gosci_Seniorow
        FOREIGN KEY (Id_Dom_Seniora) REFERENCES Domy_Seniora (Id_Domu_Seniora)
/



ALTER TABLE Seniorzy
    ADD CONSTRAINT Senior_Zajmuje_Pokoj
        FOREIGN KEY (Id_Pokoj) REFERENCES Pokoje (Id_Pokoju)
/



ALTER TABLE Karty_Zdrowia
    ADD CONSTRAINT Senior_Jest_Opisany_Przez_Zdrowie
        FOREIGN KEY (Id_Senior) REFERENCES Seniorzy (Id_Seniora)
/



ALTER TABLE Seniorzy
    ADD CONSTRAINT Senior_Ma_Adres
        FOREIGN KEY (Id_Adres) REFERENCES Adresy (Id_Adresu)
/



ALTER TABLE Pracownicy
    ADD CONSTRAINT Pracownik_Ma_Adres
        FOREIGN KEY (Id_Adres) REFERENCES Adresy (Id_Adresu)
/



ALTER TABLE Krewni
    ADD CONSTRAINT Krewny_Ma_Adres
        FOREIGN KEY (Id_Adres) REFERENCES Adresy (Id_Adresu)
/



ALTER TABLE Domy_Seniora
    ADD CONSTRAINT Dom_Seniora_Ma_Adres
        FOREIGN KEY (Id_Adres) REFERENCES Adresy (Id_Adresu)
/
```

```
ALTER TABLE Pracownicy
    ADD CONSTRAINT Pracownik_Ma_Stanowisko
        FOREIGN KEY (Id_Stanowkisko) REFERENCES Stanowiska (Id_Stanowiska)
/



ALTER TABLE Pracownicy
    ADD CONSTRAINT Zatrudnia
        FOREIGN KEY (Id_Domu_Seniora) REFERENCES Domy_Seniora (Id_Domu_Seniora)
/



ALTER TABLE Wynagrodzenia
    ADD CONSTRAINT Otrzymuje
        FOREIGN KEY (Id_Pracownika) REFERENCES Pracownicy (Id_Pracownika)
/



ALTER TABLE Krewni
    ADD CONSTRAINT Senior_Jest_Powiazany_Z_Krewnym
        FOREIGN KEY (Id_Seniora) REFERENCES Seniorzy (Id_Seniora)
/



ALTER TABLE Pracownicy_Seniorzy
    ADD CONSTRAINT Pracownicy_opiekuja_sie_seniorami
        FOREIGN KEY (Id_Seniora) REFERENCES Seniorzy (Id_Seniora)
/



ALTER TABLE Domy_Seniora
    ADD CONSTRAINT Dom_seniora_ma_sponsora
        FOREIGN KEY (Id_Sponsora) REFERENCES Sponsorzy (Id_Sponsora)
/



ALTER TABLE Czujniki_Bezpieczenstwa
    ADD CONSTRAINT Pokoj_ma_zamontowany_czujnik
        FOREIGN KEY (Id_Pokoju) REFERENCES Pokoje (Id_Pokoju)
/
```

5.4 Przykłady zapytań i poleceń SQL odnoszących się do bazy danych

Listing 2: Kod SQL przykładowych poleceń

```
   --Dodanie domu seniora
INSERT INTO Domy_Seniora
VALUES (1, 'Dom Spokojnej Jesieni', DATE '2005-04-15', 30, 50,
        'Nowoczesny osrodek dla seniorow.', 'T', 2500.00, 1, NULL);

--Dodanie seniora
INSERT INTO Seniorzy
VALUES (3, 'Katarzyna', 'Dobinska', DATE '1948-05-12', '48051212345',
        'K', DATE '2024-05-10', 'wdowa', '123789456', 'katarzyna.dobinska@gmail.com', 1, 2, 3);

--Dodanie pracownika
INSERT INTO Pracownicy
VALUES (3, 'Robert', 'Nowicki', DATE '2022-03-10', DATE '1980-04-15', 3, 1, 1);
```

```sql
--Dodanie adresu
INSERT INTO Adresy
VALUES (1, 'Krakow', 'Jesionowa', '5', '2');

INSERT INTO Adresy
VALUES (2, 'Lodz', 'Debowa', '8', '3');

--Dodanie pokoju
INSERT INTO Pokoje
VALUES (1, '101', 1, 2, 'W', 150.00, 'Pokoj dwuosobowy z balkonem.', 1);

--Aktualizacja danych

-- Zmiana numeru telefonu seniora o Id_Seniora rownym 1
UPDATE Seniorzy
SET Telefon = '789456123'
WHERE Id_Seniora = 1;

-- Aktualizacja sredniej ceny mieszkania w domu seniora o Id_Domu_Seniora rownym 1
UPDATE Domy_Seniora
SET Srednia_Cena_Mieszkania = 2700.00
WHERE Id_Domu_Seniora = 1;

--Usuwanie danych

-- Usuniecie seniora o Id_Seniora rownym 1
DELETE FROM Seniorzy
WHERE Id_Seniora = 1;

-- Usuniecie czujnika bezpieczenstwa o Id_Czujnika rownym 2
DELETE FROM Czujniki_Bezpieczenstwa
WHERE Id_Czujnika = 2;

--Pobieranie danych

-- Pobranie imienia, nazwiska, daty urodzenia i telefonu seniorow
--z domu seniora o nazwie "Dom Seniora Sloneczny"
SELECT S.Imie, S.Nazwisko, S.Data_Urodzenia, S.Telefon
SELECT S.Imie, S.Nazwisko, S.Data_Urodzenia, S.Telefon
FROM Seniorzy S
JOIN Domy_Seniora D ON S.Id_Dom_Seniora = D.Id_Domu_Seniora
WHERE D.Nazwa = 'Dom Seniora Sloneczny';

-- Pobranie numeru pokoju, pietra, liczby lozek
--oraz statusu zajetosci pokoi w domu seniora o Id_Dom_Seniora rownym 1
SELECT P.Numer_Pokoju, P.Pietro, P.Liczba_Lozek,
       CASE
           WHEN S.Id_Pokoj IS NOT NULL THEN 'Zajety'
           ELSE 'Wolny'
       END AS Status
FROM Pokoje P
LEFT JOIN Seniorzy S ON P.Id_Pokoju = S.Id_Pokoj
WHERE P.Id_Dom_Seniora = 1;

-- Pobranie danych o imieniu, nazwisku, stanie zdrowia, wadze,
--wzroscie i dacie pomiaru seniorow z domu seniora o Id_Dom_Seniora rownym 1
SELECT S.Imie, S.Nazwisko, K.Stan_Zdrowia, K.Waga, K.Wzrost, K.Data_Pomiaru
FROM Seniorzy S
JOIN Karty_Zdrowia K ON S.Id_Seniora = K.Id_Senior
WHERE S.Id_Dom_Seniora = 1;

-- Pobranie imienia, nazwiska pracownikow oraz nazwy ich stanowiska
SELECT P.Imie, P.Nazwisko, S.Nazwa AS Stanowisko
FROM Pracownicy P
JOIN Stanowiska S ON P.Id_Stanowkisko = S.Id_Stanowiska;

-- Pobranie imienia, nazwiska seniorow oraz ich wynikow cisnienia,
```

```
--jesli cisnienie gorne > 140 lub dolne > 90
SELECT S.Imie, S.Nazwisko, K.Cisnienie_Gorne, K.Cisnienie_Dolne
FROM Seniorzy S
JOIN Karty_Zdrowia K ON S.Id_Seniora = K.Id_Senior
WHERE K.Cisnienie_Gorne > 140 OR K.Cisnienie_Dolne > 90;
```