

BAZY DANYCH I BIG DATA

PROJEKT nr 1

Sprawozdanie

Temat: Spółdzielnia mieszkaniowa

Autorzy:

Maciej Kaczkowski, 300660

Paweł Berentowicz, 300481

Spis treści

1. Zakres i cel projektu	3
2. Definicja systemu	4
3. Model konceptualny.....	6
3.1. Definicja zbiorów encji określonych w projekcie oraz określenie atrybutów i ich dziedzin	6
3.2. Ustalenie związków między encjami oraz ich typów	9
3.3. Klucze kandydujące i główne.....	10
3.4. Schemat ER na poziomie konceptualnym.....	11
3.5. Pułapki szczelinowe i wachlarzowe	11
4. Model logiczny	12
4.1. Usunięcie właściwości niekompatybilnych z modelem relacyjnym	12
4.2. Proces normalizacji	13
4.2. 1. SpółdzielniaMieszkaniowa:	13
4.2. 2. StronyInternetowe:	13
4.2.3. Adresy:.....	13
4.2. 4. Zarządy:.....	13
4.2.5. CzłonkowieZarządu:.....	14
4.2.6. Pracownicy:.....	14
4.2.7. Stanowiska:.....	14

4.2.8. Wynagrodzenia:	14
4.2.9. Umowy:	14
4.2.10. Specjalizacje pracowników:	14
4.2.11. Klienci:	15
4.2.12. DziałkiBudowlane:	15
4.2.13. DomyJednorodzinne:	15
4.2.14. Bloki:	15
4.2.15. Mieszkania:	15
4.3. Schemat ER na poziomie modelu logicznego	15
4.4. Więzy integralności	15
4.5. Proces denormalizacji	16
5. Model fizyczny	17
5.1. Projekt transakcji i weryfikacja ich wykonalności.....	17
5.2. Strojanie bazy danych poprzez dobór indeksów	17
5.2.1. Klienci	17
5.2.2. Pracownicy.....	17
5.2.3. Mieszkania	17
5.3. Skrypt SQL generujący bazę danych.....	18
5.4. Przykłady zapytań i poleceń SQL odnoszących się do bazy danych	18
5.4.1. Odczytanie informacji o klientach mieszkających w danym bloku.....	18
5.4.1. Odczytanie imion i nazwisk wiceprezesów w aktualnym zarządzie	18
5.4.2. Odczytanie dat podpisania umowy oraz dat urodzenia wśród aktualnie zatrudnionych pracowników	18
6. Załączniki	19
6.1. Załącznik 1: zal_1_model_konceptualny.png	19
6.2. Załącznik 2: zal_2_model_logiczny.png.....	21
6.3. Załącznik 3: DLL_script.SQL	23
7. Bibliografia	47

1. Zakres i cel projektu

Celem pierwszej części projektu jest zaprojektowanie oraz zaimplementowanie relacyjnej bazy danych na poziomach conceptualnym i logicznym oraz jej fizyczna implementacja. Baza danych ma za zadanie umożliwić obsługę i zarządzanie spółdzielnią mieszkaniową. Zostanie ona oparta o system zarządzania bazami danych firmy Oracle.

1.1. Wykorzystane oprogramowanie

- Oracle 19c DBMS
- Oracle SQL Developer
- TOAD Data Modeler 7.2
- Git 2.31.0.windows.1

1.2. Ogólny opis

Realizowany projekt dotyczy spółdzielni mieszkaniowej. Spółdzielnia ta zajmuje się zarządzaniem danego osiedla oraz zaspokajaniem potrzeb mieszkaniowych członków i ich rodzin. Realizuje ona takie zadania jak nabywanie nowych budynków w celu ustanowienia, na rzecz członków spółdzielczych, lokatorskich lub własnościowych praw do znajdujących się w tych budynkach lokali mieszkalnych, budowanie lub nabywanie domów, lub nabywanie budynków w celu wynajmowania lub sprzedaży znajdujących się w tych budynkach lokali mieszkalnych lub lokali o innym przeznaczeniu. Zarządzanie (wynajem, remonty) już posiadanymi budynkami mieszkalnymi, nabywaniem nowych. W tym celu spółdzielnia prowadzi bazę dotyczącą mieszkań wchodzących w skład poszczególnych budynków, która jest również podstawą przy dokonywaniu wyceny mieszkania do zakupu/wynajmu.

2. Definicja systemu

2.1. Przykładowe funkcjonalności systemu

- tworzenie, modyfikacja, podgląd, usuwanie (CRUD) danych personalnych pracowników
- CRUD danych personalnych klientów
- CRUD obiektów zarządzanych przez spółdzielnię - domów, bloków, działek
- możliwość rozszerzenia struktury bazy o dodatkowe informacje (np. jeśli spółdzielnia zacznie zajmować się wynajmem bud dla psów zostanie utworzona relacja "BudaDlaPsa")

2.2. Perspektywy użytkowników

2.2.1. Administrator

Administrator ma dostęp do wszystkich funkcji systemu. Posiada uprawnienia administratora bazy danych Oracle, a zatem może modyfikować jej strukturę (np. dodawanie/ usuwanie nowych relacji), dodawać, usuwać, modyfikować i odczytywać wszystkie dane.

2.2.2. Księgowa/y

Księgowa/y ma dostęp do danych o pracownikach, o ile dotyczą one finansów. Zatem takie dane jak data podpisania umowy, data rozwiązania umowy, wynagrodzenia są dla niej/niego dostępne, a dane takie jak PESEL - nie. Oprócz tego ma dostęp do wszelkich danych związanych z kosztami i zyskami związanymi z zarządzanymi nieruchomościami oraz danymi o sobie.

2.2.3. Członek zarządu

Ma dostęp do wszystkich danych o pracownikach oraz nieruchomościach (odczytywanie, modyfikacja), ale nie może modyfikować struktury bazy danych.

2.2.4. Administrator nieruchomości

Ma dostęp do danych o sobie (modyfikacja, odczytywanie), a także do danych nieruchomości, którymi zarządza. W tym przypadku ma dostęp do wszystkich operacji CRUD - może dodawać, odczytywać, modyfikować i usuwać dane.

2.2.5. Pracownik Sprzątający

Ma dostęp do danych o sobie (modyfikacja, odczytywanie), a także do danych dotyczących bloków (ale nie jego mieszkańców), będących w posiadaniu Spółdzielni (odczytywanie).

2.2.6. Inny pracownik

Ma dostęp do danych o sobie - może je modyfikować oraz odczytywać.

2.2.7. Klient

Może odczytywać dane o sobie, a także je modyfikować. Ma dostęp do danych mieszkania, tak długo jak je zamieszkuje.

3. Model konceptualny

3.1. Definicja zbiorów encji określonych w projekcie oraz określenie atrybutów i ich dziedzin

Spółdzielnia Mieszkaniowa – główna encja					
Nazwa atrybutu	Typ i dziedzina	Czy obowiązkowy	Czy unikatowy	Czy prosty	Opis
Numer_spoldzielni_PK	SmallInt	T	T	T	Numer jednoznacznie identyfikujący spółdzielnię
Adres_spoldzielni	AdresD	T	T	N	Adres spółdzielni
REGON	Character(10)	T	T	T	Nr REGON spółdzielni
KRS	Character(9)	T	T	T	Nr KRS spółdzielni
NIP	Character(10)	T	T	T	Nr NIP spółdzielni

Zarząd – reprezentuje encję zarządzającą Spółdzielnią					
Nazwa atrybutu	Typ i dziedzina	Czy obowiązkowy	Czy unikatowy	Czy prosty	Opis
Numer_zarzadu_PK	SmallInt	T	T	T	Numer jednoznacznie identyfikujący dany zarząd
Skarbnik	SmallInt	T	T	N	Reprezentuje skarbnika zarządu
Prezes	SmallInt	T	T	N	Reprezentuje Prezesa zarządu
Vice_prezes	SmallInt	T	T	N	Reprezentuje Vice-Prezesa zarządu
Rozpoczęcie_kadencji	Date	T	N	T	Data rozpoczęcia kadencji zarządu
Planowane_zakonczenie_kadencji	Date	T	N	T	Planowana data zakończenia kadencji zarządu

Pracownik – reprezentuje encję pracownika spółdzielni					
Nazwa atrybutu	Typ i dziedzina	Czy obowiązkowy	Czy unikatowy	Czy prosty	Opis
Numer_pracownika_PK	Integer	T	T	T	Numer jednoznacznie identyfikujący danego pracownika
Pensja_miesieczna	Money	T	N	N	Reprezentuje wysokość pensji pracownika
Umowa_podpisana	Date	T	N	T	Data podpisania umowy przez pracownika
Umowa_rozwiazana	Date	N	N	T	Data rozwiązywania umowy przez pracownika
plec	PlecD	T	N	T	Płeć pracownika
PESEL	Character(11)	N	T	T	Nr PESEL pracownika
Numer_telefonu	VarChar(9)	N	T	T	Nr telefonu pracownika
Stanowisko	VarChar(512)	T	N	N	Stanowisko zajmowane przez pracownika

Adres_pracownika	AdresD	T	T	N	Adres pracownika
Data_urodzenia	Date	N	N	T	Data urodzenia pracownika
Email_pracownika	VarChar(512)	N	T	T	Adres e-mail pracownika
Imie	ImieD	T	N	T	Imię pracownika
Nazwisko	NazwiskoD	T	N	T	Nazwisko pracownika

Klient – reprezentuje encję klienta spółdzielni					
Nazwa atrybutu	Typ i dziedzina	Czy obowiązkowy	Czy unikatowy	Czy prosty	Opis
Numer_klienta_PK	Integer	T	T	T	Numer jednoznacznie identyfikujący danego klienta
Imie	ImieD	T	N	T	Imię danego klienta
Nazwisko	NazwiskoD	T	N	T	Nazwisko danego klienta
Numer_telefonu	VarChar(12)	N	N	T	Numer telefonu danego klienta
Adres_klienta	AdresD	T	T	N	Adres danego klienta
PESEL	Character(11)	N	T	T	Nr PESEL klienta
Data_urodzenia	Date	N	N	T	Data urodzenia Klienta
Od_kiedy_klient	Date	N	N	T	Data, kiedy klient stał się klientem spółdzielni
Płec	PlecD	T	N	T	Płeć klienta

Blok – reprezentuje encję bloku należącego do spółdzielni					
Nazwa atrybutu	Typ i dziedzina	Czy obowiązkowy	Czy unikatowy	Czy prosty	Opis
Numer_bloku_PK	Integer	T	T	T	Numer jednoznacznie identyfikujący dany blok
Ilosc_mieszkan	Integer	T	N	T	Ilość mieszkań znajdujących się w danym bloku
Adres_bloku	AdresD	T	T	N	Adres bloku

Mieszkanie – reprezentuje encję mieszkania zawierającego się w bloku					
Nazwa atrybutu	Typ i dziedzina	Czy obowiązkowy	Czy unikatowy	Czy prosty	Opis
Numer_mieszkania_PK	Integer	T	T	T	Numer jednoznacznie identyfikujący dane mieszkanie
Ilosc_pokoi	SmallInt	T	N	T	Ilość pokoi znajdujących się w danym mieszkaniu
Ilosc_lazienek	SmallInt	T	N	T	Ilość łazienek znajdujących się w danym mieszkaniu

Dom Jednorodzinny – reprezentuje encję domu jednorodzinnego należącego do spółdzielni					
Nazwa atrybutu	Typ i dziedzina	Czy obowiązkowy	Czy unikatowy	Czy prosty	Opis
Numer_domu_PK	Integer	T	T	T	Numer jednoznacznie identyfikujący dany dom
Powierzchnia	Float(2)	T	N	T	Powierzchnia domu
Adres_domu	AdresD	T	T	N	Adres domu

Ilosc_pieter	SmallInt	T	N	T	Ilość pięter domu
Ilosc_pokoi	SmallInt	N	N	T	Ilość pokoi w domu
Ilosc_lazienek	SmallInt	N	N	T	Ilość łazienek w domu

Działka Budowlana – reprezentuje encję działki budowlanej należącej do spółdzielni					
Nazwa atrybutu	Typ i dziedzina	Czy obowiązkowy	Czy unikatowy	Czy prosty	Opis
Numer_dzialki_PK	Integer	T	T	T	Numer jednoznacznie identyfikujący działkę
Powierzchnia	Float(10)	N	N	T	Powierzchnia działki
Adres_dzialki	AdresD	T	T	N	Adres działki
Media	MediaD	N	N	T	Media (woda, gaz, prąd)
Garaz	Boolean	N	N	T	Określa, czy działka ma garaż
Ksiega_wieczysta	Boolean	T	T	T	Określa, czy księga wieczysta działki jest aktualna

Strona Internetowa – reprezentuje encję strony internetowej należącej do spółdzielni					
Nazwa atrybutu	Typ i dziedzina	Czy obowiązkowy	Czy unikatowy	Czy prosty	Opis
Numer_strony_PK	SmallInt	T	T	T	Numer jednoznacznie identyfikujący stronę internetową
Admnistrador	VarChar(256)	T	N	N	Opisuje administratora strony internetowej
Adres_url	VarChar(256)	T	T	T	Adres URL strony internetowej

Specjalizacje Pracowników:

Pracownik Księgowości – reprezentuje encję pracownika księgowości					
Nazwa atrybutu	Typ i dziedzina	Czy obowiązkowy	Czy unikatowy	Czy prosty	Opis
Dostep_do_danych_niejawnych	VarChar(512)	T	N	N	Opisuje do jakich danych niejawnych ma dostęp dany pracownik księgowości
Numer_uprawnien	VarChar(64)	T	T	T	Numer uprawnień

Pracownik Administracji– reprezentuje encję pracownika administracji					
Nazwa atrybutu	Typ i dziedzina	Czy obowiązkowy	Czy unikatowy	Czy prosty	Opis
Zarządzane_bloki	VarChar(512)	N	N	T	Opisuje zakres zarządzania blokami
Zarządzane_domy	VarChar(512)	N	N	T	Opisuje zakres zarządzania domami

Pracownik Sprzątający– reprezentuje encję pracownika administracji					
Nazwa atrybutu	Typ i dziedzina	Czy obowiązkowy	Czy unikatowy	Czy prosty	Opis
Zarządzane_bloki	VarChar(512)	T	N	T	Opisuje sprzątane bloki

3.2. Ustalenie związków między encjami oraz ich typów

Ogólnie rzecz biorąc, w modelu bazy danych występują prawie wszystkie rodzaje związków pomiędzy encjami. Przeważające liczebnie są związki typu 1:n (jeden do wielu). Co charakterystyczne dla modelu conceptualnego - występują związki typu n:m (wiele do wielu). Są one niekompatybilne z modelem relacyjnym, jednak ich wystąpienie było spodziewane, ponieważ często oddają one związki występujące w świecie rzeczywistym. Problem niekompatybilności z modelem relacyjnym zostanie zaadresowany w dalszej części projektu. Wszystkie związki mają krotność 2.

Name	Parent Entity	Child Entity	Cardinality
Zatrudnia	SpoldzielniaMieszkaniowa	Pracownik	1...1 - 0...m
Jest_zarządzana_ przez	SpoldzielniaMieszkaniowa	Zarząd	1...1 - 1...1
Posiada_strone	SpoldzielniaMieszkaniowa	StronaInternetowa	1...1 - 0...m
Posiada_dzialke	SpoldzielniaMieszkaniowa	DzialkaBudowlana	1...1 - 0...m
Posiada_dom	SpoldzielniaMieszkaniowa	DomJednorodzinny	1...1 - 0...m
Posiada_blok	SpoldzielniaMieszkaniowa	Blok	1...1 - 0...m
Zawiera_mieszkani a	Blok	Mieszkanie	1...1 - 1...m
Zarządza_domem	AdministratorNieruchomosci	Dom	1...1 - 0...m
Zarządza_blokiem	AdministratorNieruchomosci	Blok	1...1 - 0...m
Zamieszkuje_dom	Klient	Dom	0...1 - 0...1
Zamieszkuje_mies zkanie	Klient	Blok	0...1 - 0...1
Posiada_skarbnika	PracownikKsiegowosci	Zarząd	1...1 - 1...1
Posiada_umowe	Klient	SpoldzielniaMieszk aniowa	0...n - 1...1
Sprząta_blok	PracownikSprzątajacy	Blok	1...n - 0...m
Zawiera_mieszkani a	Blok	Mieszkanie	1...1 - 1...m

3.3. Klucze kandydujące i główne

Zdecydowaliśmy się na użycie kluczy sztucznych, w celu poprawy czytelności i spójności oraz uniknięcia nietypowych błędów, które mogą się pojawić w przypadku użycia kluczy naturalnych (np. pomyłka przy wprowadzaniu numeru PESEL spowoduje złamanie zasady unikatowości kluczy). Innymi kluczami kandydującymi były numery takie jak: REGON, NIP, KRS (w przypadku spółdzielni) oraz PESEL (w przypadku ludzi). Warto jednak zauważyć, że w większości przypadków encji nie ma dobrych naturalnych kluczy kandydujących, zatem i tak jest konieczne użycie kluczy sztucznych.

Entity	Primary Key (PK)
SpoldzielniaMieszkaniowa	numer_spoldzielni_PK
Zarząd	numer_zarzadu_PK
Klient	numer_klienta_PK
Pracownik	numer_pracownika_PK
PracownikKsiegowosci	numer_pracownika_PK
Sprzedawca	numer_pracownika_PK
AdministratorNieruchomosci	numer_pracownika_PK
DzialkaBudowlana	numer_dzialki_PK
DomJednorodzinny	numer_domu_PK
Blok	numer_bloku_PK
Mieszkanie	numer_mieszkania_PK
StronaInternetowa	numer_strony_PK
Zarząd	numer_zarzadu_PK

3.4. Schemat ER na poziomie konceptualnym

Zobacz: zal_1_model_konceptualny.png

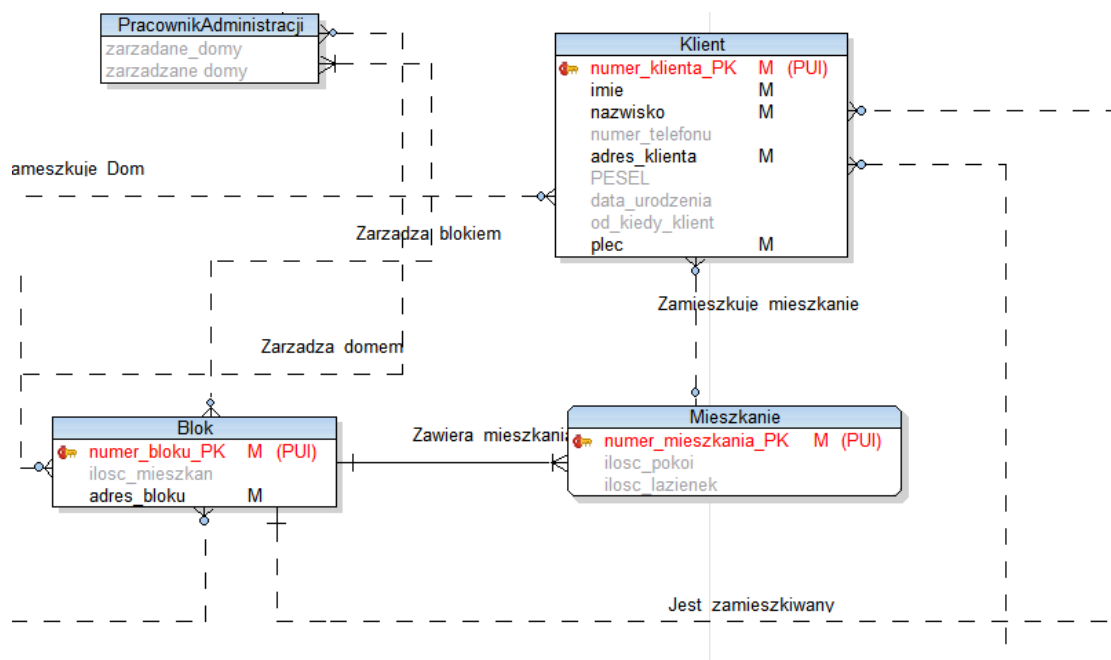
3.5. Pułapki szczelinowe i wachlarzowe

3.5.1. Wachlarzowa

W zaproponowanym modelu konceptualnym nie stwierdziliśmy nigdzie występowania problemu pułapki wachlarzowej.

3.5.2. Szczelinowa

W zaproponowanym modelu konceptualnym zauważyliśmy, że sugerowane jest istnienie związku pomiędzy zbiorami encji “Blok” i “Klient”, jednak nie istnieją żadne ścieżki łączące wystąpienia tych encji, czyli mamy do czynienia z pułapką szczelinową. Problem ten rozwiązano dodając związek pomiędzy wspomnianymi wyżej relacjami:



4. Model logiczny

4.1. Usunięcie właściwości niekompatybilnych z modelem relacyjnym

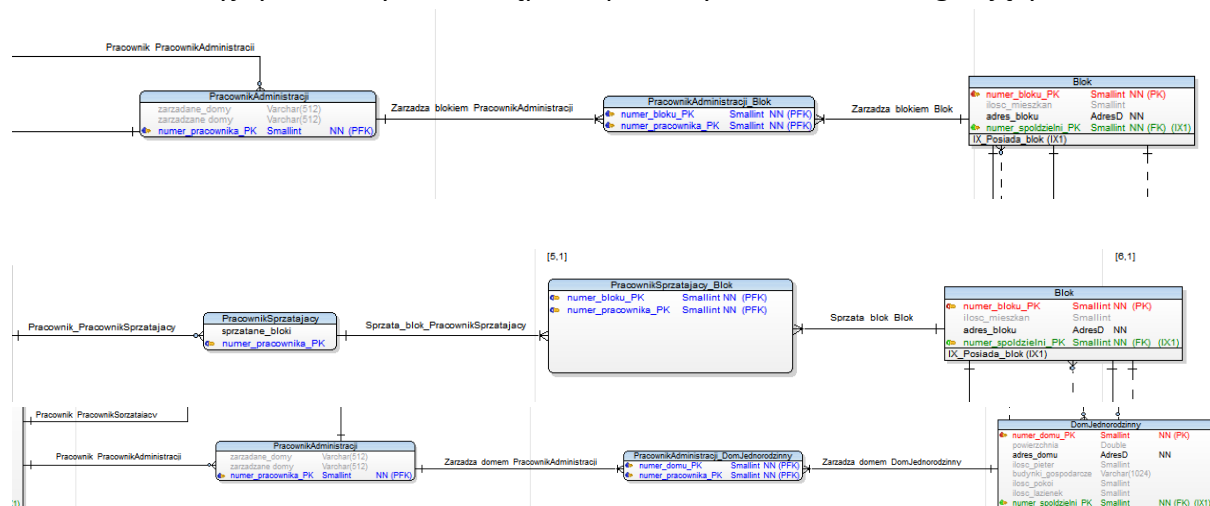
Aby przejść od zaproponowanego modelu konceptualnego do logicznego modelu relacyjnego podjęto próbę usunięcia niekompatybilności modelu konceptualnego z modelem relacyjnym poprzez następujące działania:

- związki wiele do wielu zastąpiono tzw. tablicami “bridge’ującymi”
- nazwa każdej encji została zmieniona na liczbę mnogą w celu odróżnienia relacji od encji
- identyfikujące atrybuty stały się kluczami głównymi tabeli, pozostałe atrybuty stały się niegłównymi atrybutami tabeli

W modelu konceptualnym znajdowały się 3 związki m:n (wielu do wielu):

- Sprząta_blok
- Zarządza_domem
- Zarządza_blokiem

W modelu relacyjnym zostały one zastąpione poniższymi tablicami bridge’ującymi:



4.2. Proces normalizacji

Poniżej przedstawiono zmiany, których dokonano na tym etapie względem poszczególnych relacji z modelu konceptualnego, celem usunięcia właściwości niekompatybilnych z modelem relacyjnym. Opisano m.in. zmianę atrybutów będących polami segmentowymi jako osobne relacje, modyfikację/zmianę konkretnych atrybutów itp.

4.2. 1. SpółdzielnieMieszkaniowe:

Jako, że atrybut 'Adres' jest polem segmentowym, postanowiono wyodrębnić go jako osobną relację, mając również na uwadze, że taki sam atrybut pojawia się również w innych relacjach, dzięki czemu zapewniona będzie większa wygoda i skalowalność. Po utworzeniu nowej, wspomnianej relacji usunięto pierwotny atrybut.

4.2. 2. StronyInternetowe:

Pole "Administrator" jest zarówno polem segmentowym (imie, nazwisko, data rozpoczęcia administrowania, itp.) jak i wielowartościowym (więcej niż jeden administrator) - konieczne jest stworzenie osobnej relacji "Administratorzy", gdzie zostaną wyniesione ich dane oraz linkowane z "macierzystą" relacją. Na koniec usunięto atrybut 'Administratorzy' z pierwotnej relacji.

4.2.3. Adresy:

Wszystkie atrybuty są atomowe, natomiast mamy do czynienia z powtarzającą się grupą, a mianowicie "Kod pocztowy" oraz "Poczta" (jeśli dane obiekty/ludzie są z tego samego regionu). W kontekście 1PN można się tego jeszcze nie doszukać, jednak w przypadku 3PN już tak - 'Nr_adresu' wskazuje na 'Kod_poczty', a 'Kod_poczty' wskazuje na Poczta. Dlatego powinna się tutaj pojawić dodatkowa relacja, jaką jest relacja słownikowa - "Poczty".

4.2. 4. Zarzady:

W tym przypadku widzimy, że pola "Skrabnik", "Prezes" i "Vice prezes" są polami segmentowymi (oraz wielowartościowymi), podobnie jak wcześniej w przypadku relacji 'StronyInternetowe', dlatego analogicznie zostanie utworzona tutaj nowa i linkowana relacja "CzłonekWieZarzadu".

4.2.5. Członkowie Zarządu:

Utworzone pole "Rola" wymusza albo stworzenie nowej dziedziny albo stworzenie nowej relacji słownikowej, na co zdecydowano się w tym przypadku. Utworzona relację słownikową "Rola" i usunięto następnie atrybut "Rola" w pierwotnej relacji.

4.2.6. Pracownicy:

W przypadku atrybutu "Stanowisko" występuje identyczna sytuacja jak powyżej, dlatego utworzoną nową relację słownikową (co sprzyja dynamicznemu przydzielaniu stanowisk, nie mamy na "sztywno" zadeklarowanej dziedziny), o takiej samej nazwie jak wspomniany atrybut, i dodano linkowanie do pierwotnej relacji. Pole "pensja_miesieczna" również jest pole segmentowanym, dlatego utworzono relację "Wynagrodzenie" z odpowiednimi atrybutami i dziedzinami, które linkowano następnie do tej relacji. Zamiast atrybutów "umowa podpisana" oraz "umowa rozwiązana" postanowiono stworzyć osobną relację "Umowa" - z myślą, że tak relacja będzie wykorzystana również w przypadku innych związków.

4.2.7. Stanowiska:

Relacja w takiej postaci jak pokazano spełnia już warunki 3PN.

4.2.8. Wynagrodzenia:

Relacja w takiej postaci jak pokazano spełnia już warunki 3PN.

4.2.9. Umowy:

Relacja w takiej postaci jak pokazano spełnia już warunki 3PN.

4.2.10. Specjalizacje pracowników:

Pracownicy Księgowości:

Atrybut "dostep_do_danych_niejawnych" (finansowych) jest polem segmentowym - w tym celu stworzono osobną relację "DaneNiejawne" i linkowano ją.

Pracownicy Administracji:

Relacja w takiej postaci jak pokazano spełnia już warunki 3PN.

PracownicySprzątający:

Relacja w takiej postaci jak pokazano spełnia już warunki 3PN.

4.2.11. Klienci:

Usunięto atrybut "adres_klienta" i dodano linkowanie do utworzonej wcześniej, oddzielnej relacji "Adresy". Usunięto atrybut "od_kiedy_klient", zamiast tego linkowano stworzoną na tym etapie relację "Umowy".

4.2.12. DziałkiBudowlane:

Relacja w takiej postaci jak pokazano spełnia już warunki 3PN.

4.2.13. DomyJednorodzinne:

Relacja w takiej postaci jak pokazano spełnia już warunki 3PN.

4.2.14. Bloki:

Relacja w takiej postaci jak pokazano spełnia już warunki 3PN.

4.2.15. Mieszkania:

Relacja w takiej postaci jak pokazano spełnia już warunki 3PN.

4.3. Schemat ER na poziomie modelu logicznego

Zobacz: zal_2_model_logiczny.png

4.4. Więzy integralności

Integralność zapewniono za pomocą użycia oznaczeń UNIQUE oraz NOT NULL. Klucze główne i obce, są UNIQUE NOT NULL. Inne atrybuty, których brak mógłby mieć wpływ na działanie bazy danych lub nie ma sensu logicznego (np. działka bez adresu, klient bez nazwiska) zostały oznaczone jako NOT NULL.

4.5. Proces denormalizacji

W modelu nie wykonywano denormalizacji – ze względu na brak faktycznych wymagań odnoszących się do obciążenia (np. ilość zapytań/ godzinę), a także rodzajów zapytań baza danych została utrzymana w 3PN, w celu zachowania elastyczności i możliwości rozbudowy o dalsze relacje.

5. Model fizyczny

Przed utworzeniem modelu fizycznego wykonano sprawdzenie modelu za pomocą opcji „Verify Model” oraz utworzono sekwencje *autoincrement* dla każdej głównej relacji.

5.1. Projekt transakcji i weryfikacja ich wykonalności

Transakcja	Potrzebne zasoby	Czy wykonalna
Dodawanie, modyfikacja, podgląd, usuwanie (CRUD) danych klientów	Klienci, Bloki, Umowy, Mieszkania, Adresy	Tak
CRUD danych pracowników	Pracownicy, Wynagrodzenia, Stanowiska, DaneNiejawne, Blok	Tak
CRUD danych o posiadanych przez spółdzielnię obiektach	Bloki, Domy, Mieszkania, PracownicyAdministracji, PracownicySprzątający, Adresy, Poczty	Tak
CRUD danych o spółdzielni i jej zarządzie	SpółdzielnieMieszkaniowe, Zarządy, CzłonkowieZarządu, Role	Tak

5.2. Strojanie bazy danych poprzez dobór indeksów

5.2.1. Klienci

```
CREATE INDEX IX_zamieszkuje_blok ON Klienci (numer_bloku_PK);  
CREATE INDEX IX_zamieszkuje_dom ON Klienci (numer_dom_PK);  
CREATE INDEX IX_jest_klientem_spoldzielni ON Klienci (numer_spoldzielni_PK);
```

5.2.2. Pracownicy

```
CREATE INDEX IX_pracuje_na_stanowisku ON Pracownicy (numer_stanowiska);  
CREATE INDEX IX_jest_pracownikiem_spoldzielni ON Pracownicy  
(numer_spoldzielni_PK);
```

5.2.3. Mieszkania

```
CREATE INDEX IX_jest_w_bloku ON Mieszkania (numer_bloku_PK);
```

5.3. Skrypt SQL generujący bazę danych

Zobacz: zal_3_skrypt_generacyjny.sql

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

5.4.1. Odczytanie informacji o klientach mieszkających w danym bloku

```
SELECT * FROM Klienci WHERE Nr_bloku_PK=19;
```

5.4.1. Odczytanie imion i nazwisk viceprezesów w aktualnym zarządzie

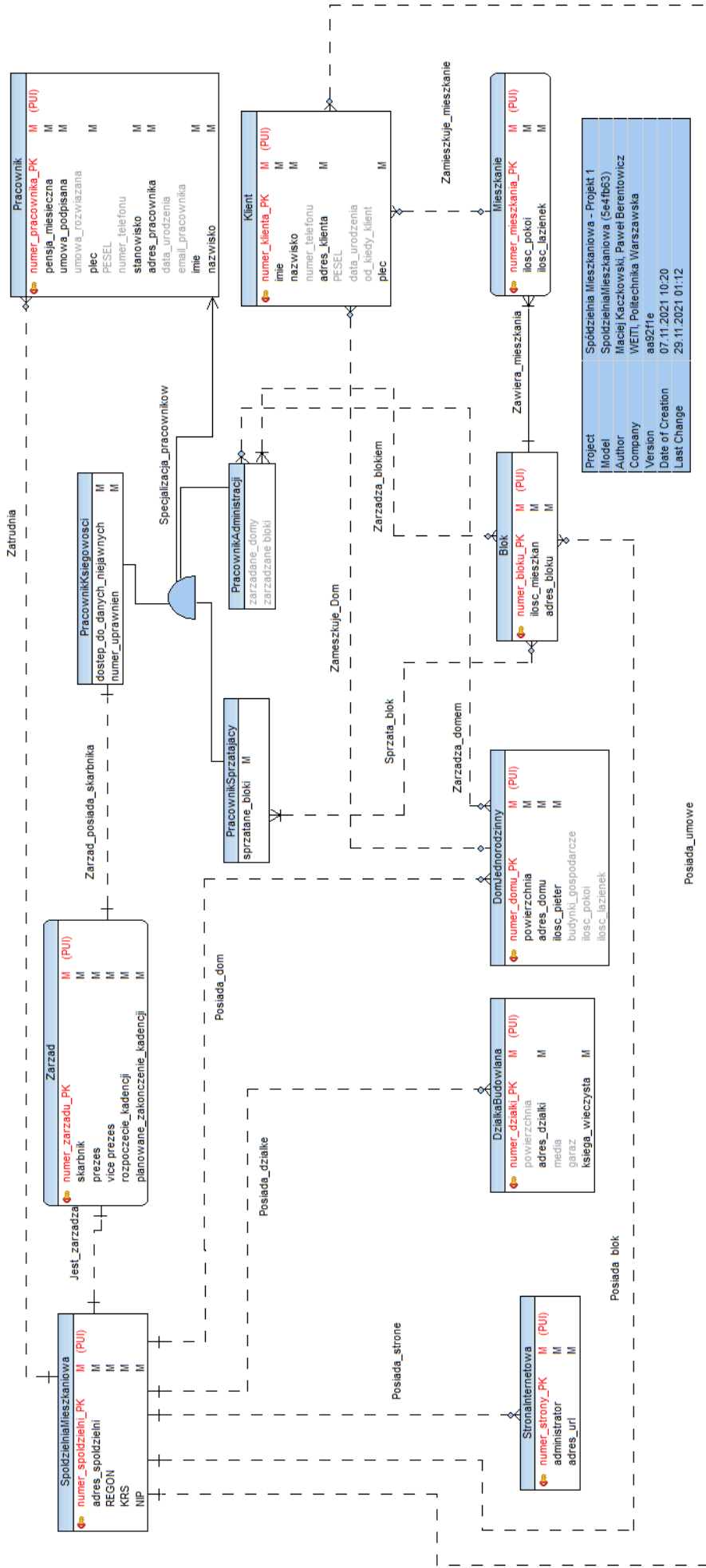
```
SELECT imie, nazwisko FROM CzlonkowieZarzadu  
JOIN Role ON CzlonkowieZarzadu.Nr_rola = Role.Nr_rola WHERE Role.Nazwa IS  
'viceprezes';
```

5.4.2. Odczytanie dat podpisania umowy oraz dat urodzenia wśród aktualnie zatrudnionych pracowników

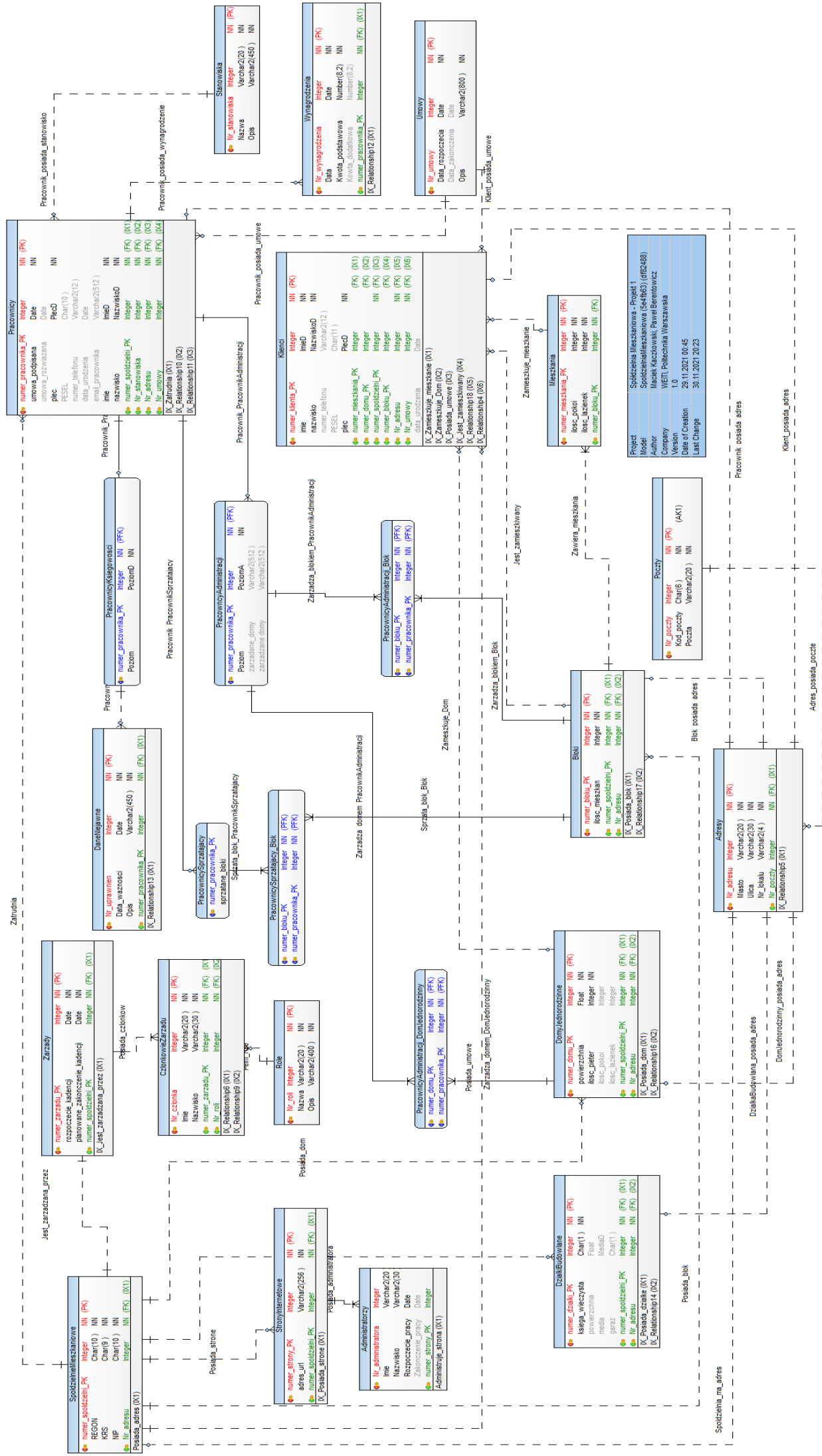
```
SELECT umowa_podpisana, data_urodzenia FROM Pracownicy WHERE  
umowa_rozwiazana IS NULL;
```

6. Załączniki

6.1. Załącznik 1: zal_1_model_konceptualny.png



6.2. Załącznik 2: zal_2_model_logiczny.png



6.3. Załącznik 3: zal_3_skrypt_generacyjny.sql

```
/*
Created: 2021-11-29
Modified: 2021-11-29
Model: SpoldzielniaMieszkaniowa (5e4fb63) (df82488)
Database: Oracle 19c
*/

-- Create sequences section -----

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

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

CREATE SEQUENCE "NrUprawnienSeq"
NOMAXVALUE
NOMINVALUE
CACHE 20
/

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

CREATE SEQUENCE "NrAdministratoraSeq"
INCREMENT BY 1
START WITH 1
NOMAXVALUE
NOMINVALUE
```

CACHE 20

/

CREATE SEQUENCE "NrStronySeq"

INCREMENT BY 1

START WITH 1

NOMAXVALUE

NOMINVALUE

CACHE 20

/

CREATE SEQUENCE "NrCzlonkaSeq"

INCREMENT BY 1

START WITH 1

NOMAXVALUE

NOMINVALUE

CACHE 20

/

CREATE SEQUENCE "NrDzialkiSeq"

INCREMENT BY 1

START WITH 1

NOMAXVALUE

NOMINVALUE

CACHE 20

/

CREATE SEQUENCE "NrDomuSeq"

INCREMENT BY 1

START WITH 1

NOMAXVALUE

NOMINVALUE

CACHE 20

/

CREATE SEQUENCE "NrBlokSeq"

INCREMENT BY 1

START WITH 1

NOMAXVALUE

NOMINVALUE

CACHE 20

/

CREATE SEQUENCE "NrAdresuSeq"

INCREMENT BY 1

START WITH 1

NOMAXVALUE

NOMINVALUE

CACHE 20

/

CREATE SEQUENCE "NrPocztySeq"

INCREMENT BY 1

START WITH 1

NOMAXVALUE

NOMINVALUE

CACHE 20

/

CREATE SEQUENCE "NrMieszkaniaSeq"

INCREMENT BY 1

START WITH 1

NOMAXVALUE

NOMINVALUE

CACHE 20

/

CREATE SEQUENCE "NrKlientaSeq"

INCREMENT BY 1

START WITH 1

NOMAXVALUE

NOMINVALUE

CACHE 20

/

CREATE SEQUENCE "NrPracownikaSeq"

INCREMENT BY 1

START WITH 1

NOMAXVALUE

NOMINVALUE

CACHE 20

/

-- Create tables section -----

-- Table SpoldzielnieMieszkaniowe

CREATE TABLE "SpoldzielnieMieszkaniowe" (

"numer_spoldzielni_PK" Integer NOT NULL,

"REGON" Char(10) NOT NULL,

"KRS" Char(9) NOT NULL,

"NIP" Char(10) NOT NULL,

"Nr_adresu" Integer NOT NULL

)

/

-- Create indexes for table SpoldzielnieMieszkaniowe

```
CREATE INDEX "IX_Relationship4" ON "SpoldzielnieMieszkaniowe" ("Nr_adresu")
/
```

-- Add keys for table SpoldzielnieMieszkaniowe

```
ALTER TABLE "SpoldzielnieMieszkaniowe" ADD CONSTRAINT "numer_spoldzielni_PK"
PRIMARY KEY ("numer_spoldzielni_PK")
/
```

-- Table Zarzady

```
CREATE TABLE "Zarzady"(
  "numer_zarzadu_PK" Integer NOT NULL,
  "rozpoczecie_kadencji" Date NOT NULL,
  "planowane_zakonczenie_kadencji" Date NOT NULL,
  "numer_spoldzielni_PK" Integer NOT NULL
)
/
```

-- Create indexes for table Zarzady

```
CREATE INDEX "IX_Jest_zarządzana_przez" ON "Zarzady" ("numer_spoldzielni_PK")
/
```

-- Add keys for table Zarzady

```
ALTER TABLE "Zarzady" ADD CONSTRAINT "numer_zarzadu_PK" PRIMARY KEY
("numer_zarzadu_PK")
/
```

-- Table Bloki

```
CREATE TABLE "Bloki"(
  "numer_bloku_PK" Integer NOT NULL,
  "ilosc_mieszkan" Integer NOT NULL,
  "numer_spoldzielni_PK" Integer NOT NULL,
  "Nr_adresu" Integer NOT NULL
)
/
```

-- Create indexes for table Bloki

```
CREATE INDEX "IX_Posiada_blok" ON "Bloki" ("numer_spoldzielni_PK")
/
```

```
CREATE INDEX "IX_Relationship17" ON "Bloki" ("Nr_adresu")
```

/

-- Add keys for table Bloki

```
ALTER TABLE "Bloki" ADD CONSTRAINT "numer_bloku_PK" PRIMARY KEY
("numer_bloku_PK")
```

/

-- Table Mieszkania

```
CREATE TABLE "Mieszkania"(
  "numer_mieszkania_PK" Integer NOT NULL,
  "ilosc_pokoi" Integer NOT NULL,
  "ilosc_lazienek" Integer NOT NULL,
  "numer_bloku_PK" Integer NOT NULL
```

)

/

-- Add keys for table Mieszkania

```
ALTER TABLE "Mieszkania" ADD CONSTRAINT "numer_mieszkani_PK" PRIMARY KEY
("numer_mieszkania_PK")
```

/

-- Table DzialkiBudowlane

```
CREATE TABLE "DzialkiBudowlane"(
  "numer_dzialki_PK" Integer NOT NULL,
  "ksiega_wieczysta" Char(1 ) NOT NULL,
  "powierzchnia" Float,
  "media" Varchar2(256 ),
  "garaz" Char(1 ),
  "numer_spoldzielni_PK" Integer NOT NULL,
  "Nr_adresu" Integer NOT NULL
```

)

/

-- Create indexes for table DzialkiBudowlane

```
CREATE INDEX "IX_Posiada_dzialke" ON "DzialkiBudowlane" ("numer_spoldzielni_PK")
```

/

```
CREATE INDEX "IX_Relationship14" ON "DzialkiBudowlane" ("Nr_adresu")
```

/

-- Add keys for table DzialkiBudowlane

```
ALTER TABLE "DzialkiBudowlane" ADD CONSTRAINT "numer_dzialki_PK" PRIMARY KEY
("numer_dzialki_PK")
/
```

-- Table Pracownicy

```
CREATE TABLE "Pracownicy"(
  "numer_pracownika_PK" Integer NOT NULL,
  "umowa_podpisana" Date NOT NULL,
  "umowa_rozwiazana" Date,
  "plec" Char(1 ) NOT NULL,
  "PESEL" Char(10 ),
  "numer_telefonu" Varchar2(12 ),
  "data_urodzenia" Date,
  "email_pracownika" Varchar2(512 ),
  "imie" Varchar2(64 ) NOT NULL,
  "nazwisko" Varchar2(64 ) NOT NULL,
  "numer_spoldzielni_PK" Integer NOT NULL,
  "Nr_stanowiska" Integer NOT NULL,
  "Nr_adresu" Integer NOT NULL,
  "Nr_umowy" Integer NOT NULL
)
/
```

-- Create indexes for table Pracownicy

```
CREATE INDEX "IX_Zatrudnia" ON "Pracownicy" ("numer_spoldzielni_PK")
/
```

```
CREATE INDEX "IX_Relationship10" ON "Pracownicy" ("Nr_stanowiska")
/
```

```
CREATE INDEX "IX_Relationship11" ON "Pracownicy" ("Nr_adresu")
/
```

```
CREATE INDEX "IX_Relationship3" ON "Pracownicy" ("Nr_umowy")
/
```

-- Add keys for table Pracownicy

```
ALTER TABLE "Pracownicy" ADD CONSTRAINT "numer_pracownika_PK" PRIMARY KEY
("numer_pracownika_PK")
/
```

-- Table Klienci

```
CREATE TABLE "Klienci"(
  "numer_klienta_PK" Integer NOT NULL,
```

```

"imie" Varchar2(64 ) NOT NULL,
"nazwisko" Varchar2(64 ) NOT NULL,
"numer_telefonu" Varchar2(12 ),
"PESEL" Char(11 ),
"plec" Char(1 ) NOT NULL,
"numer_mieszkania_PK" Integer,
"numer_domu_PK" Integer,
"numer_spoldzielni_PK" Integer NOT NULL,
"numer_bloku_PK" Integer,
"Nr_adresu" Integer NOT NULL,
"Nr_umowy" Integer NOT NULL,
"data_urodzenia" Date
)
/

-- Create indexes for table Klienci

CREATE INDEX "IX_Zamieszkuje_mieszkanie" ON "Klienci" ("numer_mieszkania_PK")
/

CREATE INDEX "IX_Zamieszkuje_Dom" ON "Klienci" ("numer_domu_PK")
/

CREATE INDEX "IX_Posiada_umowe" ON "Klienci" ("numer_spoldzielni_PK")
/

CREATE INDEX "IX_Jest_zamieszkiwany" ON "Klienci" ("numer_bloku_PK")
/

CREATE INDEX "IX_Relationship18" ON "Klienci" ("Nr_adresu")
/

CREATE INDEX "IX_Relationship4" ON "Klienci" ("Nr_umowy")
/

-- Add keys for table Klienci

ALTER TABLE "Klienci" ADD CONSTRAINT "numer_klienta_PK" PRIMARY KEY
("numer_klienta_PK")
/

-- Table StronyInternetowe

CREATE TABLE "StronyInternetowe"(
"numer_strony_PK" Integer NOT NULL,
"adres_url" Varchar2(256 ) NOT NULL,
"numer_spoldzielni_PK" Integer NOT NULL
)

```

/

-- Create indexes for table StronyInternetowe

```
CREATE INDEX "IX_Posiada_strone" ON "StronyInternetowe" ("numer_spoldzielni_PK")
```

/

-- Add keys for table StronyInternetowe

```
ALTER TABLE "StronyInternetowe" ADD CONSTRAINT "numer_strony_PK" PRIMARY KEY ("numer_strony_PK")
```

/

-- Table DomyJednorodzinne

```
CREATE TABLE "DomyJednorodzinne"(  
  "numer_domu_PK" Integer NOT NULL,  
  "powierzchnia" Float NOT NULL,  
  "ilosc_pieter" Integer NOT NULL,  
  "ilosc_pokoi" Integer,  
  "ilosc_lazienek" Integer,  
  "numer_spoldzielni_PK" Integer NOT NULL,  
  "Nr_adresu" Integer NOT NULL
```

)

/

-- Create indexes for table DomyJednorodzinne

```
CREATE INDEX "IX_Posiada_dom" ON "DomyJednorodzinne" ("numer_spoldzielni_PK")
```

/

```
CREATE INDEX "IX_Relationship16" ON "DomyJednorodzinne" ("Nr_adresu")
```

/

-- Add keys for table DomyJednorodzinne

```
ALTER TABLE "DomyJednorodzinne" ADD CONSTRAINT "numer_domu_PK" PRIMARY KEY ("numer_domu_PK")
```

/

-- Table PracownicySprzatajacy

```
CREATE TABLE "PracownicySprzatajacy"(  
  "numer_pracownika_PK" Integer NOT NULL,  
  "sprzatane_bloki" Varchar2(512 ) NOT NULL
```

)

/

-- Add keys for table PracownicySprzatajacy

```
ALTER TABLE "PracownicySprzatajacy" ADD CONSTRAINT "Unique_Identifier5" PRIMARY  
KEY ("numer_pracownika_PK")  
/
```

-- Table PracownicyKsiegowosci

```
CREATE TABLE "PracownicyKsiegowosci"(  
  "numer_pracownika_PK" Integer NOT NULL,  
  "Poziom" Varchar2(30 ) NOT NULL  
)  
/
```

-- Add keys for table PracownicyKsiegowosci

```
ALTER TABLE "PracownicyKsiegowosci" ADD CONSTRAINT "Unique_Identifier6" PRIMARY  
KEY ("numer_pracownika_PK")  
/
```

-- Table and Columns comments section

```
COMMENT ON COLUMN "PracownicyKsiegowosci"."Poziom" IS 'Określa stopień Księgowego  
(Młodszy/Zwykły/Starszy/Główny Księgowy)'  
/
```

-- Table PracownicyAdministracji

```
CREATE TABLE "PracownicyAdministracji"(  
  "numer_pracownika_PK" Integer NOT NULL,  
  "Poziom" Varchar2(30 ) NOT NULL,  
  "zarzadane_domy" Varchar2(512 ),  
  "zarzadzane domy" Varchar2(512 )  
)  
/
```

-- Add keys for table PracownicyAdministracji

```
ALTER TABLE "PracownicyAdministracji" ADD CONSTRAINT "Unique_Identifier7" PRIMARY  
KEY ("numer_pracownika_PK")  
/
```

-- Table and Columns comments section

```
COMMENT ON COLUMN "PracownicyAdministracji"."Poziom" IS 'Mówi o Poziomie Pracownika  
Administracji.'  
/
```

-- Table PracownicyAdministracji_Blok

```
CREATE TABLE "PracownicyAdministracji_Blok"(  
  "numer_bloku_PK" Integer NOT NULL,  
  "numer_pracownika_PK" Integer NOT NULL  
)  
/
```

-- Table PracownicyAdministracji_DomJednorodzinny

```
CREATE TABLE "PracownicyAdministracji_DomJednorodzinny"(  
  "numer_domu_PK" Integer NOT NULL,  
  "numer_pracownika_PK" Integer NOT NULL  
)  
/
```

-- Table PracownicySprzatajacy_Blok

```
CREATE TABLE "PracownicySprzatajacy_Blok"(  
  "numer_bloku_PK" Integer NOT NULL,  
  "numer_pracownika_PK" Integer NOT NULL  
)  
/
```

-- Table Adresy

```
CREATE TABLE "Adresy"(  
  "Nr_adresu" Integer NOT NULL,  
  "Miasto" Varchar2(20 ) NOT NULL,  
  "Ulica" Varchar2(30 ) NOT NULL,  
  "Nr_lokalu" Varchar2(4 ) NOT NULL,  
  "Nr_poczty" Integer NOT NULL  
)  
/
```

-- Create indexes for table Adresy

```
CREATE INDEX "IX_Relationship5" ON "Adresy" ("Nr_poczty")  
/
```

-- Add keys for table Adresy

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

-- Table and Columns comments section


```
COMMENT ON COLUMN "Adresy"."Nr_adresu" IS 'Unikatowy identyfikator adresu (klucz  
główny).'
```

```
/
```

```
COMMENT ON COLUMN "Adresy"."Miasto" IS 'Miasto'
```

```
/
```

```
COMMENT ON COLUMN "Adresy"."Ulica" IS 'Ulica'
```

```
/
```

```
COMMENT ON COLUMN "Adresy"."Nr_lokalu" IS 'Numer lokalu.'
```

```
/
```

```
-- Table Administratorzy
```

```
CREATE TABLE "Administratorzy"(  
  "Nr_administratora" Integer NOT NULL,  
  "Imie" Varchar2(20 ) NOT NULL,  
  "Nazwisko" Varchar2(30 ) NOT NULL,  
  "Rozpoczecie_pracy" Date NOT NULL,  
  "Zakonczenie_pracy" Date,  
  "numer_strony_PK" Integer NOT NULL
```

```
)
```

```
/
```

```
-- Create indexes for table Administratorzy
```

```
CREATE INDEX "IX_Relationship3" ON "Administratorzy" ("numer_strony_PK")
```

```
/
```

```
-- Add keys for table Administratorzy
```

```
ALTER TABLE "Administratorzy" ADD CONSTRAINT "PK_Administratorzy" PRIMARY KEY  
("Nr_administratora")
```

```
/
```

```
-- Table and Columns comments section
```

```
COMMENT ON TABLE "Administratorzy" IS 'Relacja Administratorzy'
```

```
/
```

```
COMMENT ON COLUMN "Administratorzy"."Nr_administratora" IS 'Unikatowy identyfikator  
Administratora.'
```

```
/
```

```
COMMENT ON COLUMN "Administratorzy"."Imie" IS 'Imię Administratora'
```

```
/
```

```
COMMENT ON COLUMN "Administratorzy"."Nazwisko" IS 'nazwisko Administratora'
```

```
/
```

```
COMMENT ON COLUMN "Administratorzy"."Rozpoczecie_pracy" IS 'Data rozpoczęcia pracy jako  
Administrator.'
```

```
/
```

```
COMMENT ON COLUMN "Administratorzy"."Zakonczenie_pracy" IS 'Data zakończenia pracy jako Administrator'
/
```

```
-- Table Poczty
```

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

```
-- Add keys for table Poczty
```

```
ALTER TABLE "Poczty" ADD CONSTRAINT "PK_Poczty" PRIMARY KEY ("Nr_poczty")
/
```

```
ALTER TABLE "Poczty" ADD CONSTRAINT "Kod_poczty" UNIQUE ("Kod_poczty")
/
```

```
-- Table and Columns comments section
```

```
COMMENT ON TABLE "Poczty" IS 'Relacja słownikowa poczty.'
/
COMMENT ON COLUMN "Poczty"."Nr_poczty" IS 'Unikatowy identyfikator poczty'
/
COMMENT ON COLUMN "Poczty"."Kod_poczty" IS 'Kod danej poczty '
/
COMMENT ON COLUMN "Poczty"."Poczta" IS 'Lokalizacja poczty'
/
```

```
-- Table CzlonkowieZarzadu
```

```
CREATE TABLE "CzlonkowieZarzadu"(
  "Nr_czlonka" Integer NOT NULL,
  "Imie" Varchar2(20 ) NOT NULL,
  "Nazwisko" Varchar2(30 ) NOT NULL,
  "numer_zarzadu_PK" Integer NOT NULL,
  "Nr_rol" Integer NOT NULL
)
/
```

```
-- Create indexes for table CzlonkowieZarzadu
```

```
CREATE INDEX "IX_Relationship6" ON "CzlonkowieZarzadu" ("numer_zarzadu_PK")
/
```

```
CREATE INDEX "IX_Relationship9" ON "CzlonkowieZarzadu" ("Nr_rol")  
/
```

```
-- Add keys for table CzlonkowieZarzadu
```

```
ALTER TABLE "CzlonkowieZarzadu" ADD CONSTRAINT "PK_CzlonkowieZarzadu" PRIMARY  
KEY ("Nr_czlonka")  
/
```

```
-- Table and Columns comments section
```

```
COMMENT ON TABLE "CzlonkowieZarzadu" IS 'Relacja Czlonkowie Zarządu. '  
/
```

```
COMMENT ON COLUMN "CzlonkowieZarzadu"."Nr_czlonka" IS 'Unikatowy identyfikator  
członka zarządu'  
/
```

```
COMMENT ON COLUMN "CzlonkowieZarzadu"."Imie" IS 'Imię członka zarządu'  
/
```

```
COMMENT ON COLUMN "CzlonkowieZarzadu"."Nazwisko" IS 'Nazwisko członka zarządu'  
/
```

```
-- Table Role
```

```
CREATE TABLE "Role"(  
  "Nr_rol" Integer NOT NULL,  
  "Nazwa" Varchar2(20 ) NOT NULL,  
  "Opis" Varchar2(400 ) NOT NULL  
)  
/
```

```
-- Add keys for table Role
```

```
ALTER TABLE "Role" ADD CONSTRAINT "PK_Role" PRIMARY KEY ("Nr_rol")  
/
```

```
-- Table and Columns comments section
```

```
COMMENT ON TABLE "Role" IS 'Opisuje pełnioną rolę przez Członka Zarządu.'  
/
```

```
COMMENT ON COLUMN "Role"."Nr_rol" IS 'Unikalny identyfikator pełnionej roli.'  
/
```

```
COMMENT ON COLUMN "Role"."Nazwa" IS 'Nazwa pełnionej roli'  
/
```

```
COMMENT ON COLUMN "Role"."Opis" IS 'Opis pełnionej roli'  
/
```

```
-- Table Stanowiska
```

```

CREATE TABLE "Stanowiska"(
  "Nr_stanowiska" Integer NOT NULL,
  "Nazwa" Varchar2(20 ) NOT NULL,
  "Opis" Varchar2(450 ) NOT NULL
)
/

-- Add keys for table Stanowiska

ALTER TABLE "Stanowiska" ADD CONSTRAINT "PK_Stalowiska" PRIMARY KEY
("Nr_stanowiska")
/

-- Table and Columns comments section

COMMENT ON COLUMN "Stanowiska"."Nr_stanowiska" IS 'Unikalny identyfikator stanowiska'
/
COMMENT ON COLUMN "Stanowiska"."Nazwa" IS 'Nazwa pelnionego stanowiska'
/
COMMENT ON COLUMN "Stanowiska"."Opis" IS 'Opis pelnionego stanowiska'
/

-- Table Wynagrodzenia

CREATE TABLE "Wynagrodzenia"(
  "Nr_wynagrodzenia" Integer NOT NULL,
  "Data" Date NOT NULL,
  "Kwota_podstawowa" Number(8,2) NOT NULL,
  "Kowta_dodatkowa" Number(8,2),
  "numer_pracownika_PK" Integer NOT NULL
)
/

-- Create indexes for table Wynagrodzenia

CREATE INDEX "IX_Relationship12" ON "Wynagrodzenia" ("numer_pracownika_PK")
/

-- Add keys for table Wynagrodzenia

ALTER TABLE "Wynagrodzenia" ADD CONSTRAINT "PK_Wynagrodzenia" PRIMARY KEY
("Nr_wynagrodzenia")
/

-- Table and Columns comments section

COMMENT ON TABLE "Wynagrodzenia" IS 'Wynagrodzenie pracownika'
/

```

```
COMMENT ON COLUMN "Wynagrodzenia"."Nr_wynagrodzenia" IS 'Unikalny identyfikator
wynagrodzenia'
/
COMMENT ON COLUMN "Wynagrodzenia"."Data" IS 'Data wypłaty wynagrodzenia dla
pracownika'
/
COMMENT ON COLUMN "Wynagrodzenia"."Kwota_podstawowa" IS 'Wartość wypłacanego
wynagrodzenia podstawowego'
/
COMMENT ON COLUMN "Wynagrodzenia"."Kowta_dodatkowa" IS 'Wartość wypłacanej kwoty
dodatkowej'
/
```

-- Table DaneNiejawne

```
CREATE TABLE "DaneNiejawne"(
  "Nr_uprawnien" Integer NOT NULL,
  "Data_waznosci" Date NOT NULL,
  "Opis" Varchar2(450 ) NOT NULL,
  "numer_pracownika_PK" Integer NOT NULL
)
/
```

-- Create indexes for table DaneNiejawne

```
CREATE INDEX "IX_Relationship13" ON "DaneNiejawne" ("numer_pracownika_PK")
/
```

-- Add keys for table DaneNiejawne

```
ALTER TABLE "DaneNiejawne" ADD CONSTRAINT "PK_DaneNiejawne" PRIMARY KEY
("Nr_uprawnien")
/
```

-- Table and Columns comments section

```
COMMENT ON TABLE "DaneNiejawne" IS 'Relacja opisująca Dane Niejawne (finansowe)
niedostępne dla pozostałych pracowników.'
/
COMMENT ON COLUMN "DaneNiejawne"."Nr_uprawnien" IS 'Unikatowy identyfikator upranień
do danych niejawnych Księgowego'
/
COMMENT ON COLUMN "DaneNiejawne"."Data_waznosci" IS 'Data ważności uzyskanych
uprawnień'
/
COMMENT ON COLUMN "DaneNiejawne"."Opis" IS 'Opis uzyskanych upranwnień, do jakich
danych ma dostęp (finansowych)'
/
```

-- Table Umowy

```
CREATE TABLE "Umowy"(  
  "Nr_umowy" Integer NOT NULL,  
  "Data_roz poczenia" Date NOT NULL,  
  "Data_zakonczenia" Date,  
  "Opis" Varchar2(800 ) NOT NULL  
)  
/
```

-- Add keys for table Umowy

```
ALTER TABLE "Umowy" ADD CONSTRAINT "PK_Umowy" PRIMARY KEY ("Nr_umowy")  
/
```

-- Table and Columns comments section

```
COMMENT ON COLUMN "Umowy"."Nr_umowy" IS 'Unikatowy numer identyfikujący umowę'  
/  
COMMENT ON COLUMN "Umowy"."Data_roz poczenia" IS 'Data wejścia umowy w życie'  
/  
COMMENT ON COLUMN "Umowy"."Data_zakonczenia" IS 'Data zakończenia obowiązywania  
umowy'  
/  
COMMENT ON COLUMN "Umowy"."Opis" IS 'Treść umowy, precyzująca jej rodzaj, formę,  
warunki itp.'  
/
```

-- Trigger for sequence NrSpoldzielniSeq for column numer_spoldzielni_PK in table
SpoldzielnieMieszkaniowe -----

```
CREATE OR REPLACE TRIGGER "ts_SpoldzielnieMieszkaniowe_NrSpoldzielniSeq" BEFORE  
INSERT
```

```
ON "SpoldzielnieMieszkaniowe" FOR EACH ROW  
BEGIN
```

```
  :new."numer_spoldzielni_PK" := "NrSpoldzielniSeq".nextval;  
END;
```

/

```
CREATE OR REPLACE TRIGGER "tsu_SpoldzielnieMieszkaniowe_NrSpoldzielniSeq" AFTER  
UPDATE OF "numer_spoldzielni_PK"
```

```
ON "SpoldzielnieMieszkaniowe" FOR EACH ROW  
BEGIN
```

```
  RAISE_APPLICATION_ERROR(-20010,'Cannot update column "numer_spoldzielni_PK" in table  
"SpoldzielnieMieszkaniowe" as it uses sequence.');
```

```
END;
```

/

-- Trigger for sequence NrZarzaduSeq for column numer_zarzadu_PK in table Zarzady -----

```

CREATE OR REPLACE TRIGGER "ts_Zarzady_NrZarzaduSeq" BEFORE INSERT
ON "Zarzady" FOR EACH ROW
BEGIN
    :new."numer_zarzadu_PK" := "NrZarzaduSeq".nextval;
END;
/

CREATE OR REPLACE TRIGGER "tsu_Zarzady_NrZarzaduSeq" AFTER UPDATE OF
"numer_zarzadu_PK"
ON "Zarzady" FOR EACH ROW
BEGIN
    RAISE_APPLICATION_ERROR(-20010,'Cannot update column "numer_zarzadu_PK" in table
"Zarzady" as it uses sequence.');
```

END;

/

```

-- Trigger for sequence NrBlokSeq for column numer_bloku_PK in table Bloki -----
CREATE OR REPLACE TRIGGER "ts_Bloki_NrBlokSeq" BEFORE INSERT
ON "Bloki" FOR EACH ROW
BEGIN
    :new."numer_bloku_PK" := "NrBlokSeq".nextval;
END;
/

CREATE OR REPLACE TRIGGER "tsu_Bloki_NrBlokSeq" AFTER UPDATE OF
"numer_bloku_PK"
ON "Bloki" FOR EACH ROW
BEGIN
    RAISE_APPLICATION_ERROR(-20010,'Cannot update column "numer_bloku_PK" in table
"Bloki" as it uses sequence.');
```

END;

/

```

-- Trigger for sequence NrMieszkaniaSeq for column numer_mieszkania_PK in table Mieszkania ----
----
CREATE OR REPLACE TRIGGER "ts_Mieszkania_NrMieszkaniaSeq" BEFORE INSERT
ON "Mieszkania" FOR EACH ROW
BEGIN
    :new."numer_mieszkania_PK" := "NrMieszkaniaSeq".nextval;
END;
/

CREATE OR REPLACE TRIGGER "tsu_Mieszkania_NrMieszkaniaSeq" AFTER UPDATE OF
"numer_mieszkania_PK"
ON "Mieszkania" FOR EACH ROW
BEGIN
    RAISE_APPLICATION_ERROR(-20010,'Cannot update column "numer_mieszkania_PK" in table
"Mieszkania" as it uses sequence.');
```

END;

/

```

-- Trigger for sequence NrDzialkiSeq for column numer_dzialki_PK in table DzialkiBudowlane -----
--
CREATE OR REPLACE TRIGGER "ts_DzialkiBudowlane_NrDzialkiSeq" BEFORE INSERT
ON "DzialkiBudowlane" FOR EACH ROW
BEGIN
    :new."numer_dzialki_PK" := "NrDzialkiSeq".nextval;
END;
/

CREATE OR REPLACE TRIGGER "tsu_DzialkiBudowlane_NrDzialkiSeq" AFTER UPDATE OF
"numer_dzialki_PK"
ON "DzialkiBudowlane" FOR EACH ROW
BEGIN
    RAISE_APPLICATION_ERROR(-20010,'Cannot update column "numer_dzialki_PK" in table
"DzialkiBudowlane" as it uses sequence.');
```

```

END;
/

-- Trigger for sequence NrPracownikaSeq for column numer_pracownika_PK in table Pracownicy ----
-----
CREATE OR REPLACE TRIGGER "ts_Pracownicy_NrPracownikaSeq" BEFORE INSERT
ON "Pracownicy" FOR EACH ROW
BEGIN
    :new."numer_pracownika_PK" := "NrPracownikaSeq".nextval;
END;
/

CREATE OR REPLACE TRIGGER "tsu_Pracownicy_NrPracownikaSeq" AFTER UPDATE OF
"numer_pracownika_PK"
ON "Pracownicy" FOR EACH ROW
BEGIN
    RAISE_APPLICATION_ERROR(-20010,'Cannot update column "numer_pracownika_PK" in table
"Pracownicy" as it uses sequence.');
```

```

END;
/

-- Trigger for sequence NrKlientaSeq for column numer_klienta_PK in table Klienci -----
CREATE OR REPLACE TRIGGER "ts_Klienci_NrKlientaSeq" BEFORE INSERT
ON "Klienci" FOR EACH ROW
BEGIN
    :new."numer_klienta_PK" := "NrKlientaSeq".nextval;
END;
/

CREATE OR REPLACE TRIGGER "tsu_Klienci_NrKlientaSeq" AFTER UPDATE OF
"numer_klienta_PK"
ON "Klienci" FOR EACH ROW
BEGIN
    RAISE_APPLICATION_ERROR(-20010,'Cannot update column "numer_klienta_PK" in table
"Klienci" as it uses sequence.');
```

```

END;
/

```



```

/

-- Trigger for sequence NrStronySeq for column numer_strony_PK in table StronyInternetowe -----
-
CREATE OR REPLACE TRIGGER "ts_StronyInternetowe_NrStronySeq" BEFORE INSERT
ON "StronyInternetowe" FOR EACH ROW
BEGIN
    :new."numer_strony_PK" := "NrStronySeq".nextval;
END;
/

CREATE OR REPLACE TRIGGER "tsu_StronyInternetowe_NrStronySeq" AFTER UPDATE OF
"numer_strony_PK"
ON "StronyInternetowe" FOR EACH ROW
BEGIN
    RAISE_APPLICATION_ERROR(-20010,'Cannot update column "numer_strony_PK" in table
"StronyInternetowe" as it uses sequence.');
```

```

END;
/

-- Trigger for sequence NrDomuSeq for column numer_domu_PK in table DomyJednorodzinne -----
--
CREATE OR REPLACE TRIGGER "ts_DomyJednorodzinne_NrDomuSeq" BEFORE INSERT
ON "DomyJednorodzinne" FOR EACH ROW
BEGIN
    :new."numer_domu_PK" := "NrDomuSeq".nextval;
END;
/

CREATE OR REPLACE TRIGGER "tsu_DomyJednorodzinne_NrDomuSeq" AFTER UPDATE OF
"numer_domu_PK"
ON "DomyJednorodzinne" FOR EACH ROW
BEGIN
    RAISE_APPLICATION_ERROR(-20010,'Cannot update column "numer_domu_PK" in table
"DomyJednorodzinne" as it uses sequence.');
```

```

END;
/

-- Trigger for sequence NrAdresuSeq for column Nr_adresu in table Adresy -----
CREATE OR REPLACE TRIGGER "ts_Adresy_NrAdresuSeq" BEFORE INSERT
ON "Adresy" FOR EACH ROW
BEGIN
    :new."Nr_adresu" := "NrAdresuSeq".nextval;
END;
/

CREATE OR REPLACE TRIGGER "tsu_Adresy_NrAdresuSeq" 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 NrAdministratoraSeq for column Nr_administratora in table Administratorzy -
-----
CREATE OR REPLACE TRIGGER "ts_Administratorzy_NrAdministratoraSeq" BEFORE INSERT
ON "Administratorzy" FOR EACH ROW
BEGIN
    :new."Nr_administratora" := "NrAdministratoraSeq".nextval;
END;
/

CREATE OR REPLACE TRIGGER "tsu_Administratorzy_NrAdministratoraSeq" AFTER UPDATE
OF "Nr_administratora"
ON "Administratorzy" FOR EACH ROW
BEGIN
    RAISE_APPLICATION_ERROR(-20010,'Cannot update column "Nr_administratora" in table
"Administratorzy" as it uses sequence.');
```

```

END;
/

-- Trigger for sequence NrPocztySeq for column Nr_poczty in table Poczty -----
CREATE OR REPLACE TRIGGER "ts_Poczty_NrPocztySeq" BEFORE INSERT
ON "Poczty" FOR EACH ROW
BEGIN
    :new."Nr_poczty" := "NrPocztySeq".nextval;
END;
/

CREATE OR REPLACE TRIGGER "tsu_Poczty_NrPocztySeq" 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 NrCzlonkaSeq for column Nr_czlonka in table CzlonkowieZarzadu -----
CREATE OR REPLACE TRIGGER "ts_CzlonkowieZarzadu_NrCzlonkaSeq" BEFORE INSERT
ON "CzlonkowieZarzadu" FOR EACH ROW
BEGIN
    :new."Nr_czlonka" := "NrCzlonkaSeq".nextval;
END;
/

CREATE OR REPLACE TRIGGER "tsu_CzlonkowieZarzadu_NrCzlonkaSeq" AFTER UPDATE
OF "Nr_czlonka"
ON "CzlonkowieZarzadu" FOR EACH ROW
BEGIN
    RAISE_APPLICATION_ERROR(-20010,'Cannot update column "Nr_czlonka" in table
"CzlonkowieZarzadu" as it uses sequence.');
```

```

END;
/

-- Trigger for sequence NrRoliSeq for column Nr_rol in table Role -----
CREATE OR REPLACE TRIGGER "ts_Role_NrRoliSeq" BEFORE INSERT
ON "Role" FOR EACH ROW
BEGIN
    :new."Nr_rol" := "NrRoliSeq".nextval;
END;
/

CREATE OR REPLACE TRIGGER "tsu_Role_NrRoliSeq" AFTER UPDATE OF "Nr_rol"
ON "Role" FOR EACH ROW
BEGIN
    RAISE_APPLICATION_ERROR(-20010,'Cannot update column "Nr_rol" in table "Role" as it uses
sequence.');
```

```

END;
/

-- Trigger for sequence NrUprawnienSeq for column Nr_uprawnien in table DaneNiejawne -----
CREATE OR REPLACE TRIGGER "ts_DaneNiejawne_NrUprawnienSeq" BEFORE INSERT
ON "DaneNiejawne" FOR EACH ROW
BEGIN
    :new."Nr_uprawnien" := "NrUprawnienSeq".nextval;
END;
/

CREATE OR REPLACE TRIGGER "tsu_DaneNiejawne_NrUprawnienSeq" AFTER UPDATE OF
"Nr_uprawnien"
ON "DaneNiejawne" FOR EACH ROW
BEGIN
    RAISE_APPLICATION_ERROR(-20010,'Cannot update column "Nr_uprawnien" in table
"DaneNiejawne" as it uses sequence.');
```

```

END;
/

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

ALTER TABLE "Zarzady" ADD CONSTRAINT "Jest_zaradzana_przez" FOREIGN KEY
("numer_spoldzielni_PK") REFERENCES "SpoldzielnieMieszkaniowe" ("numer_spoldzielni_PK")
/

ALTER TABLE "StronyInternetowe" ADD CONSTRAINT "Posiada_strone" FOREIGN KEY
("numer_spoldzielni_PK") REFERENCES "SpoldzielnieMieszkaniowe" ("numer_spoldzielni_PK")
/

```

```
ALTER TABLE "Klienci" ADD CONSTRAINT "Zamieszkuje_mieszkanie" FOREIGN KEY  
("numer_mieszkania_PK") REFERENCES "Mieszkania" ("numer_mieszkania_PK")  
/
```

```
ALTER TABLE "DzialkiBudowlane" ADD CONSTRAINT "Posiada_dzialke" FOREIGN KEY  
("numer_spoldzielni_PK") REFERENCES "SpoldzielnieMieszkaniowe" ("numer_spoldzielni_PK")  
/
```

```
ALTER TABLE "DomyJednorodzinne" ADD CONSTRAINT "Posiada_dom" FOREIGN KEY  
("numer_spoldzielni_PK") REFERENCES "SpoldzielnieMieszkaniowe" ("numer_spoldzielni_PK")  
/
```

```
ALTER TABLE "Pracownicy" ADD CONSTRAINT "Zatrudnia" FOREIGN KEY  
("numer_spoldzielni_PK") REFERENCES "SpoldzielnieMieszkaniowe" ("numer_spoldzielni_PK")  
/
```

```
ALTER TABLE "Mieszkania" ADD CONSTRAINT "Zawiera_mieszkania" FOREIGN KEY  
("numer_bloku_PK") REFERENCES "Blok" ("numer_bloku_PK")  
/
```

```
ALTER TABLE "Blok" ADD CONSTRAINT "Posiada_blok" FOREIGN KEY  
("numer_spoldzielni_PK") REFERENCES "SpoldzielnieMieszkaniowe" ("numer_spoldzielni_PK")  
/
```

```
ALTER TABLE "Klienci" ADD CONSTRAINT "Zameszkuje_Dom" FOREIGN KEY  
("numer_domu_PK") REFERENCES "DomyJednorodzinne" ("numer_domu_PK")  
/
```

```
ALTER TABLE "Klienci" ADD CONSTRAINT "Posiada_umowe" FOREIGN KEY  
("numer_spoldzielni_PK") REFERENCES "SpoldzielnieMieszkaniowe" ("numer_spoldzielni_PK")  
/
```

```
ALTER TABLE "Klienci" ADD CONSTRAINT "Jest_zamieszkiwany" FOREIGN KEY  
("numer_bloku_PK") REFERENCES "Blok" ("numer_bloku_PK")  
/
```

```
ALTER TABLE "Administratorzy" ADD CONSTRAINT "Posiada_administratora" FOREIGN KEY  
("numer_strony_PK") REFERENCES "StronyInternetowe" ("numer_strony_PK")  
/
```

```
ALTER TABLE "SpoldzielnieMieszkaniowe" ADD CONSTRAINT "Spoldzielnia_ma_adres"  
FOREIGN KEY ("Nr_adresu") REFERENCES "Adresy" ("Nr_adresu")  
/
```

```
ALTER TABLE "Adresy" ADD CONSTRAINT "Adres_posiada_poczte" FOREIGN KEY  
("Nr_poczty") REFERENCES "Poczty" ("Nr_poczty")  
/
```

```
ALTER TABLE "CzlonkowieZarzadu" ADD CONSTRAINT "Posiada_czlonkow" FOREIGN KEY  
("numer_zarzadu_PK") REFERENCES "Zarzady" ("numer_zarzadu_PK")  
/
```

```
ALTER TABLE "CzlonkowieZarzadu" ADD CONSTRAINT "Pelni_role" FOREIGN KEY  
("Nr_rol") REFERENCES "Role" ("Nr_rol")  
/
```

```
ALTER TABLE "Pracownicy" ADD CONSTRAINT "Pracownik_posiada_stanowisko" FOREIGN  
KEY ("Nr_stanowiska") REFERENCES "Stanowiska" ("Nr_stanowiska")  
/
```

```
ALTER TABLE "Pracownicy" ADD CONSTRAINT "Pracownik_posiada_adres" FOREIGN KEY  
("Nr_adresu") REFERENCES "Adresy" ("Nr_adresu")  
/
```

```
ALTER TABLE "Wynagrodzenia" ADD CONSTRAINT "Pracownik_posiada_wynagrodzenie"  
FOREIGN KEY ("numer_pracownika_PK") REFERENCES "Pracownicy"  
("numer_pracownika_PK")  
/
```

```
ALTER TABLE "DaneNiejawne" ADD CONSTRAINT  
"PracownikKsiegowosci_posiada_dostep_do_danych_niejawnych" FOREIGN KEY  
("numer_pracownika_PK") REFERENCES "PracownicyKsiegowosci" ("numer_pracownika_PK")  
/
```

```
ALTER TABLE "DzialkiBudowlane" ADD CONSTRAINT "DzialkaBudowlana_posiada_adres"  
FOREIGN KEY ("Nr_adresu") REFERENCES "Adresy" ("Nr_adresu")  
/
```

```
ALTER TABLE "DomyJednorodzinne" ADD CONSTRAINT "DomJednorodzinny_posiada_adres"  
FOREIGN KEY ("Nr_adresu") REFERENCES "Adresy" ("Nr_adresu")  
/
```

```
ALTER TABLE "Bloki" ADD CONSTRAINT "Blok_posiada_adres" FOREIGN KEY ("Nr_adresu")  
REFERENCES "Adresy" ("Nr_adresu")  
/
```

```
ALTER TABLE "Klienci" ADD CONSTRAINT "Klient_posiada_adres" FOREIGN KEY  
("Nr_adresu") REFERENCES "Adresy" ("Nr_adresu")  
/
```

```
ALTER TABLE "Pracownicy" ADD CONSTRAINT "Pracownik_posiada_umowe" FOREIGN KEY  
("Nr_umowy") REFERENCES "Umowy" ("Nr_umowy")  
/
```

```
ALTER TABLE "Klienci" ADD CONSTRAINT "Klient_posiada_umowe" FOREIGN KEY  
("Nr_umowy") REFERENCES "Umowy" ("Nr_umowy")  
/
```

7. Bibliografia

- Wykład i materiały wykładowe z przedmiotu „Bazy Danych i Big Data” prowadzonego przez dr hab. inż. Marcina Kowalczyka w semestrze zimowym 2021
- Dokumentacja programów Oracle19c, SQL Developer oraz TOAD Data Modeller