Dokumentacja projektu z BDBT
Dom Kultury

Sofiia Levchenko 308996, Mateusz Izbicki 303957

Politechnika Warszawska, Wydział Elektroniki i Technik Informacyjnych

12 lipca 2022

Spis treści

1 Zakres i cel projektu

Celem naszego projektu jest zbudowanie systemu bazy danych dla Domu Kultury. Zrealizowanie projektu obejmuje:

- Specyfikacja wymagań
- Stworzenie modelu konceptualnego
- Przejście do modelu logicznego
- Zapewnienie trzeciej postaci normalnej modelu logicznego
- Wygenerowanie kodu reprezentującego stworzonej bazy danych w języku SQL
- Wypełnienie bazy danych danymi

Podstawowymi funkcjonalnościami naszego systemu będą:

- Zarządzanie zasobami w salach (ile krzeseł jest w danej sali itp.)
- Zarządzanie wydarzeniami organizowanych w ośrodku
- Zapisywanie się uczestników na organizowane wydarzenia

2 Definicja systemu

2.1 Zakres bazy danych

Opis działalności biznesowej:

Realizowany projekt dotyczy przedsiębiorstwa z branży społeczno-naukowej. Przedsiębiorstwo to zajmuje się organizacją i przeprowadzeniem warsztatów oraz wystaw różnego zakresu na terenie Domu Kultury.

W swojej codziennej działalności dom kultury skupia się na przeprowadzeniu warsztatów oraz wystaw, organizowanych przez pracowników danej jednostki. Wydarzenia te mogą trwać dowolną liczbę godzin w przedziale czasowym między otwarciem a zamknięciem jednostki. Na każde wydarzenie może się zapisać dowolna ilość osób, żeby zapisać się osoba chętna musi podać dane osobowe oraz dane do skontaktowania się. Na podstawie działalności przeprowadzanej przez dom kultury tworzony jest również publicznie dostępny harmonogram wydarzeń, czyli w jakiej sali odbywa się, w jakim odcinku czasowym. Również każdy użytkownik przed zapisaniem się na zajęcia może zapoznać się ze szczegółami przeprowadzanych wydarzeń, czyli jego typ, imię oraz nazwisko wystawiającego lub wykładowcy (w zależności od typu wydarzenia, nie są pracownikami domu kultury), temat oraz ewentualnie opis wydarzenia. Wszystkie wydarzenia mają odpowiedzialnych za niego osób, czyli pracowników domu kultury które organizują te wydarzenie. Wszystkie wydarzenia na terenie Domu Kultury prowadzone są za darmo, wymagana jest tylko wcześniejsze zapisanie się.

2.2 Specyfikacja bazy danych

Wymagania funkcjonalne i niefunkcjonalne, **założenia co do wymagań funkcjonalnych obsługiwanych przez system:**

- Przyjęcie przez system danych osobowych użytkowników chcących zarejestrować się na zajęcia

  Funkcja ta umożliwia sprawdzenie poprawności wpisania wszystkich podanych informacji dotyczących danych osobowych, sprawdzenie poprawności wskazanych danych kontaktowych (np odesłania na wskazany mail z kod, po wpisaniu którego w odpowiednim miejscu użytkownik zostaje zidentyfikowany).

- Przypisanie użytkownika do listy osób, będących uczestnikami wydarzeń

  Każda osoba, która wpisała dane osobowe oraz potwierdziła poprawność wpisanych przez nią danych kontaktowych jest dodawana do bazy danych jako uczestnik wydarzenia.

- Harmonogram zajęć w postaci kalendarza

  Wszystkie wydarzenia organizowane przez dom kultury będą umieszczone w szczególnym harmonogramie czasu, który będzie umożliwiał udostępnienie wykładowcom, wystawiającym oraz wszystkim chętnym zobaczyć wszystkie wydarzenia odbywające się we wszystkich salach w ośrodku. Harmonogram będzie pozwalał na bezproblemowe przypisanie wydarzeniu konkretnej sali bez wystąpienia konfliktów (np dwa wydarzenia przypisane są do tej samej sali w tym samym czasie), zmiany sali lub godziny w których odbywa się wydarzenie

- Lista osób zapisanych na zajęcia

  Pracownicy ośrodku oraz wykładowcy będą mieli dostęp do informacji osobowej uczestników konkretnego wydarzenia (jest przydatne w razie anulowania wydarzenia, zmiany czasu przeprowadzenia, żeby informować uczestników o wszelkich zmianach)

- Informacja dotycząca sali w której prowadzone są zajęcia

  W celu sensownego przypisania sali do wydarzenia pracownicy ośrodku mają dostęp do informacji dotyczących wszystkich sal (jaki jest jej rozmiar, jakie zasoby posiada), żeby zapewnić najlepszą kompatybilność pomiędzy przeprowadzanymi zajęciami a salą w której one są prowadzane (np dopasowanie się do ilości zapisanych osób)

- Dodatkowa informacja dotycząca organizowanych wydarzeń

  Każda chętna osoba (pracownicy ośrodku, prowadzący zajęć, osoba wystawiająca, wszyscy chętni do zapisania się na zajęcia, uczestnicy wydarzenia) ma dostęp do wszelkich informacji dotyczących bezpośrednio wydarzenia, czyli godziny w których odbywają się, ile osób już zapisały się, numer sali itp

- Informacja o pracownikach organizujących wydarzenia

  Każde wydarzenie posiada osób organizujących konkretne wydarzenie, te osoby kontrolują wszystkie szczegóły dotyczące dostarczania jakichś dodatkowych informacji (ewentualnie o wystąpienie jakichkolwiek zmian w trybie prowadzenia wydarzeń) do uczestników wydarzeń oraz prowadzących lub wystawiających

2.3 Perspektywy użytkowników i zidentyfikowane transakcje

Perspektywy użytkowników:

- **Kierownik domu kultury**

  Mający dostęp do wszystkich danych

- **Pracownik**

  Mający dostęp do swoich danych osobowych, danych potrzebne do wykonania zadań które obejmuje posiadanie przez niego stanowisko

- **Uczestnik wydarzenia**

  Będzie posiadał dostęp do zobaczenia na jakie wydarzenia się zapisał i ich szczegółów

Zidentyfikowane transakcje:

- w obszarze obsługa pracowników
  - dodawanie i usuwanie pracowników
  - modyfikacja danych osobowych
  - przypisanie pracownika do opiekowaniem sie nad wydarzeniem
  - nadawanie stanowisk pracownikom
- w obszarze domu kultury
  - zdefiniowanie domu kultury
  - modyfikowanie podstawowych wartości atrybutów
- w obszarze obsługa wydarzeń
  - tworzenie i usuwanie wydarzeń
  - zarządzanie uczestnikami który zapisali się na dane wydarzenie
  - modyfikacja informacji o wydarzeniu
- w obszarze uczestnik wydarzenia
  - dodanie i usuwanie uczestników wydarzeń
  - informowanie o zmianach w trybie przeprowadzenia wydarzeń do których ten uczestnik jest przypisany
- w obszarze logistyka
  - dodawanie i usuwanie zasobów z poszczególnych sal
  - zakup i wyrzucenie zasobów
  - modyfikacja informacji o zasobach

3 Model konceptualny

3.1 Definicja zbiorów encji określonych w projekcie

- Dom Kultury - encja główna, zawierająca podstawowe informacje na temat Domu Kultury
- Pracownik - encja zawierająca informacje o pracowniku
- Sala - encja zawierająca informacje o sali znajdującej się w domu kultury
- Wydarzenie - encja zawierająca informacje o wydarzeniu odbywającym się w ośrodku
- Warsztat - encja rozszerzająca encję Wydarzenia, która posiada informacje o specyficznym wydarzeniu jakim jest warsztat
- Wystawa - encja rozszerzająca encje Wydarzenia, w której zawierają się informacje o wystawie
- Zasób - encja zawierająca informację o posiadanym przez dom kultury zasobie wykorzystywanym do prowadzenia zajęć
- Uczestnik Wydarzenia - encja zawierająca informację o zarejestrowanym na wydarzenie użytkownikowi

3.2 Ustawienie związków między encjami i ich typów

Nazwa Encji	**Typ relacji**	**Opis**	**Nazwa Encji**	**Typ relacji**	**Opis**
Dom Kultury	1..1	Zatrudnia	Pracownik	0..*	Jest zatrudniany
Dom Kultury	1..1	Posiada	Sala	0..*	Znajduje się
Dom Kultury	1..1	Organizuje	Wydarzenie	0..*	Jest organizowana
Sala	0..*	Korzysta	Zasób	0..*	Jest wykorzystywany
Sala	0..*	Jest używana	Wydarzenie	0..*	Używa
Uczestnik wydarzenia	0..*	Uczestniczy	Wydarzenie	0..*	Ma
Wydarzenie	x	Rozszerza się	Warsztat	x	Rozszerzenie
Wydarzenie	x	Rozszerza się	Wystawa	x	Rozszerzenie

3.3 Określenie atrybutów i ich dziedzin

Nazwa Encji: Dom Kultury						
Nazwa atrybutu	**Opis atrybutu**	**Dziedzina atrybutu**	**Czy obowiązkowy**	**Klucz główny**	**Czy atrybut prosty**	**Dodatkowe informacje**
Id_domu_kultury	Unikatowy numer domu kultury	Integer	T	T	T	
Telefon	Nr telefonu do Domu Klutury	VarChar(12)	T	N	N	Pole zgodne z jedną z masek numerów telefonicznych
Adres	Adres domu kultury	VarChar (200)	T	N	N	Pole segmentowe, obejmujące takie informacje jak miasto, ulica, numer lokalizacji, kod pocztowy, poczta
Godzina_otwarcia	Godzina o której otwierany jest dom kultury	Time	T	N	T	
Godzina_zamkniecia	Godzina o której zamykany jest dom kultury	Time	T	N	T	

Nazwa Encji: Sala						
Nazwa atrybutu	**Opis atrybutu**	**Dziedzina atrybutu**	**Czy obowiązkowy**	**Klucz główny**	**Czy atrybut prosty**	**Dodatkowe informacje**
Id_sali	Unikalny identyfikator sali znajdującej sie w domie kultury	Integer	T	T	T	
Powierzchnia	Metraż sali(w m^2)	Float(2)	T	N	T	
Numer_sali	Numer sali w domu kultury (np. 102)	VarChar (5)	T	N	T	

Nazwa Encji: Zasób						
Nazwa atrybutu	**Opis atrybutu**	**Dziedzina atrybutu**	**Czy obowiązkowy**	**Klucz główny**	**Czy atrybut prosty**	**Dodatkowe informacje**
Id_zasobu	Unikalny identyfikator zasobu znajdującego sie w sali	Integer	T	T	T	
Nazwa	Nazwa zasobu (nadana przez producenta)	VarChar (30)	T	N	T	
Producent	Nazwa producenta zasobu	VarChar (30)	T	N	T	
Data_kupna	Data kupna zasobu	Date	T	N	T	

Nazwa Encji: Pracownik						
Nazwa atrybutu	**Opis atrybutu**	**Dziedzina atrybutu**	**Czy obowiązkowy**	**Klucz główny**	**Czy atrybut prosty**	**Dodatkowe informacje**
Id_pracownika	Unikalny identyfikator pracownika	Integer	T	T	T	
Nazwisko	Nazwisko pracownika		T	N	T	
Imie	Imie pracownika		T	N	T	
PESEL	Identyfikator osobowy z Powszechnej Spisu Ewidencji Ludności	Character(11)	N	N	T	Dopuszcza sie wartość null gdy pracownik nie posiada PESELu (pracownik z zagranicy)
Stanowisko	Stanowisko pracownika	VarChar (30)	T	N	T	
Data_urodzenia	Data urodzenia pracownika	Date	T	N	T	
Adres	Adres pracownika	VarChar(200)	T	N	N	Pole segmentowe, obejmujące takie informacje jak miasto, ulica, numer lokalizacji, kod pocztowy, poczta
Telefon	Telefon pracownika	VarChar (12)	N	N	N	Pole wielowartościowe.Zgodny z jedną z masek numerów telefonicznych
Plec	Płeć pracownika	ENUM {K,M}	T	N	T	
Pensja	Miesięczna pensja pracownika	Number	T	N	T	

Nazwa Encji: Wydarzenie						
Nazwa atrybutu	**Opis atrybutu**	**Dziedzina atrybutu**	**Czy obowiązkowy**	**Klucz główny**	**Czy atrybut prosty**	**Dodatkowe informacje**
Id_wydarzenia	Unikalny identyfikator wydarzenia	Integer	T	T	T	
Typ_wydarzenia	Typ wydarzenia warsztat albo wystawa	ENUM {warsztat, wystawa}	T	N	T	
Data	Data w której odbywa sie wydarzenie	Date	T	N	T	
Czas_trwania	Czas trwania wydarzenia w pełnych godzinach	Integer	T	N	T	

Nazwa Encji: Uczestnik Wydarzenia						
Nazwa atrybutu	**Opis atrybutu**	**Dziedzina atrybutu**	**Czy obowiązkowy**	**Klucz główny**	**Czy atrybut prosty**	**Dodatkowe informacje**
Id_uczestnika	Unikalny identyfikator uczestnika biorącego udział w wydarzeniu	Integer	T	T	T	
Imie	Imię uczestnika	VarChar(30)	T	N	T	
Nazwisko	Nazwisko uczestnika	VarChar(30)	T	N	T	
Telefon	Telefon do uczestnika	VarChar(12)	T	N	N	Pole wielowartościowe.Zgodny z jedną z masek numerów telefonicznych.
Email	Email do uczestnika	VarChar(30)	T	N	T	Pole wymagane aby miec jakikolwiek kontakt z uczestnikiem w wypadku odwołania wydarzenia

Nazwa Encji: Warsztat						
Nazwa atrybutu	**Opis atrybutu**	**Dziedzina atrybutu**	**Czy obowiązkowy**	**Klucz główny**	**Czy atrybut prosty**	**Dodatkowe informacje**
Temat	Temat warsztatu	VarChar (150)	T	N	T	
Imie_wykladowcy	Imię wykladowcy warsztatu	VarChar (30)	T	N	T	
Nazwisko_wykladowcy	Nazwisko wykladowcy warsztatu	VarChar (30)	T	N	T	
Telefon	Telefon do wykladowcy warsztatu	VarChar (12)	N	N	N	Pole wielowartościowe. Zgodny z jedną z masek numerów telefonicznych.
Email	Email do wykładowcy	VarChar (30)	T	N	T	Pole wymagane aby miec jakikolwiek kontakt z wykladowcą warsztatu

Nazwa Encji: Wystawa						
Nazwa atrybutu	**Opis atrybutu**	**Dziedzina atrybutu**	**Czy obowiązkowy**	**Klucz główny**	**Czy atrybut prosty**	**Dodatkowe informacje**
Temat	Temat wystawy	VarChar (150)	T	N	T	
Typ_wystawy	Typ wystawy	ENUM { malarska, fotograficzna, interaktywna, muzyczna, filmowa }	T	N	T	
Imie_wystawiajacego	Imię wystawiającego wystawe	VarChar (30)	T	N	T	
Nazwisko_wystawiajacego	Nazwisko wystawiającego wystawe	VarChar (30)	T	N	T	
Telefon	Telefon do wystawiającego wystawe	VarChar (12)	T	N	N	Pole wielowartościowe. Zgodny z jedną z masek numerów telefonicznych.
Email	Email do wystawiającego	VarChar (30)	T	N	T	Pole wymagane aby miec jakikolwiek kontakt z wystawiającym warsztatu
Opis	Opis wystawy	VarChar (400)	N	N	T	

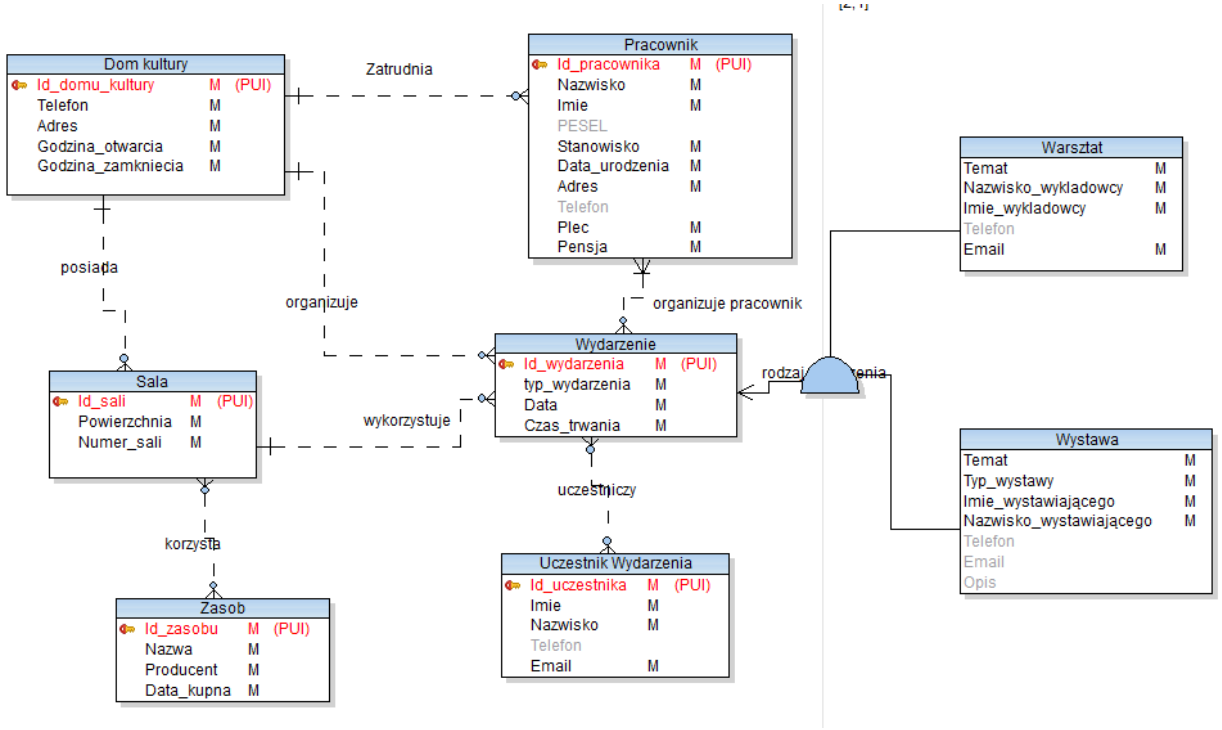
3.4 Dodatkowe reguły integralnościowe

Dom {}Kultury ⟷ Pracownik - wyciąganie informacji na temat pracowników w ośrodku
Dom Kultury ⟷ - pobieranie informacji na temat sal w Domie Kultury
Dom Kultury ⟷ Wydarzenie - wyciąganie informacji o wydarzeniach które mają miejsce w domu kultury
Sala ⟷ Zasoby - pobieranie informacji i ilości zasobów w danej sali
Wydarzenie ⟷ Uczestnik Wydarzenia - wyciąganie informacji o ilości uczestników
wydarzenie, rejestracja użytkowników

3.5 Klucze kandydujące i główne

Nazwa Encji	Klucz główny	Klucze kandydujące
Dom Kultury	Id_domu_kultury	Adres, Telefon
Pracownik	Id_pracownika	Imię, Nazwisko, Data_urodzenie, Stanowisko, Adres, Plec
Sala	Id_sali	Numer sali
Wydarzenie	Id_wydarzania	Data
Zasób	Id_zasobu	Nazwa
Uczestnik Wydarzenia	Id_uczestnika	Imię, Nazwisko

3.6 Schemat ER na poziomie konceptualnym



Rys. 1: Schemat modelu ER na poziomie konceptualnym

Po zaprojektowaniu modelu konceptualnego za pomocą narzędzia TOAD Data Modeler sprawdziliśmy poprawność modelu poprzez wbudowaną opcję 'Verify', po której uruchomieniu nie dostaliśmy żadnych błędów krytycznych, czyli zakładamy, że nasz model jest poprawny.

3.7 Problem pułapek szczelinowych i wachlarzowych

W monecie projektowania modelu mieliśmy w głowie to aby przestrzegać się tego aby nie wystąpiły owe pułapki więc nie mieliśmy dużo do naprawiania, jednak jeden przykład możemy zaprezentować.

Problemem było wystąpienie pułapki szczelinowej w przypadku połączenia encji Wydarzenie z Domem kultury. Problem polegał na tym że nie było relacji pomiędzy tymi encjami więc nie mogliśmy jednoznacznie wyszukać wszystkich wydarzeń w domie kultury bo bez tej relacji musieliśmy przejść przez encję Pracownik która nie jednoznacznie określała dane wydarzenie w Domu Kultury.

4 Model logiczny

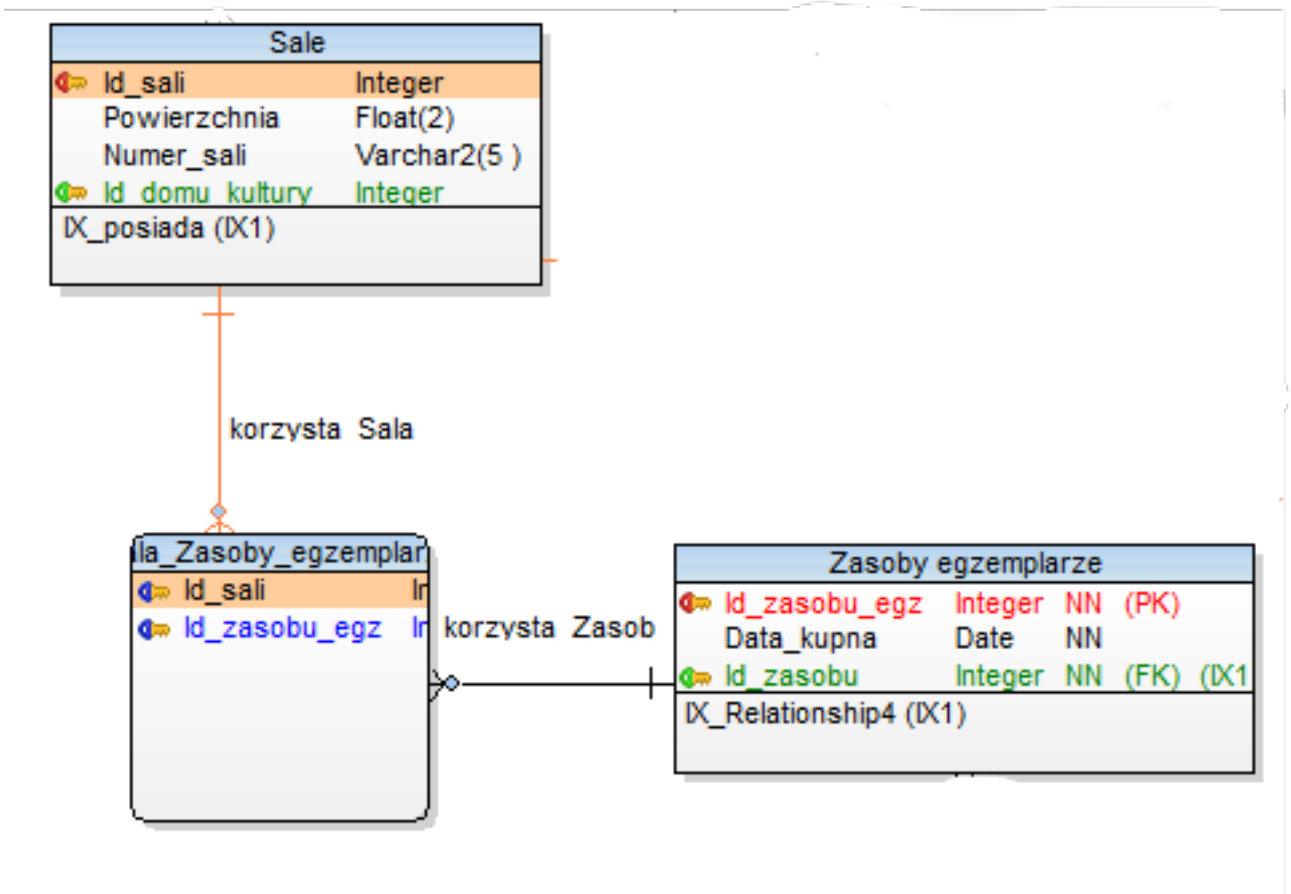
4.1 Charakterystyka modelu relacyjnego

Projekt relacyjnej bazy danych polega na znalezieniu właściwych schematów relacji tworzących bazę danych. Żeby nasz projekt relacyjnej bazy danych spełniał te warunki zajęliśmy się tym, że:

- pozbyliśmy się redundancji danych
- zapewniliśmy reprezentowania związków między danymi, poszczególnymi encjami
- zachowywaliśmy warunki integralności, umożliwiliśmy kontrole warunków integralności podczas modyfikacji danych

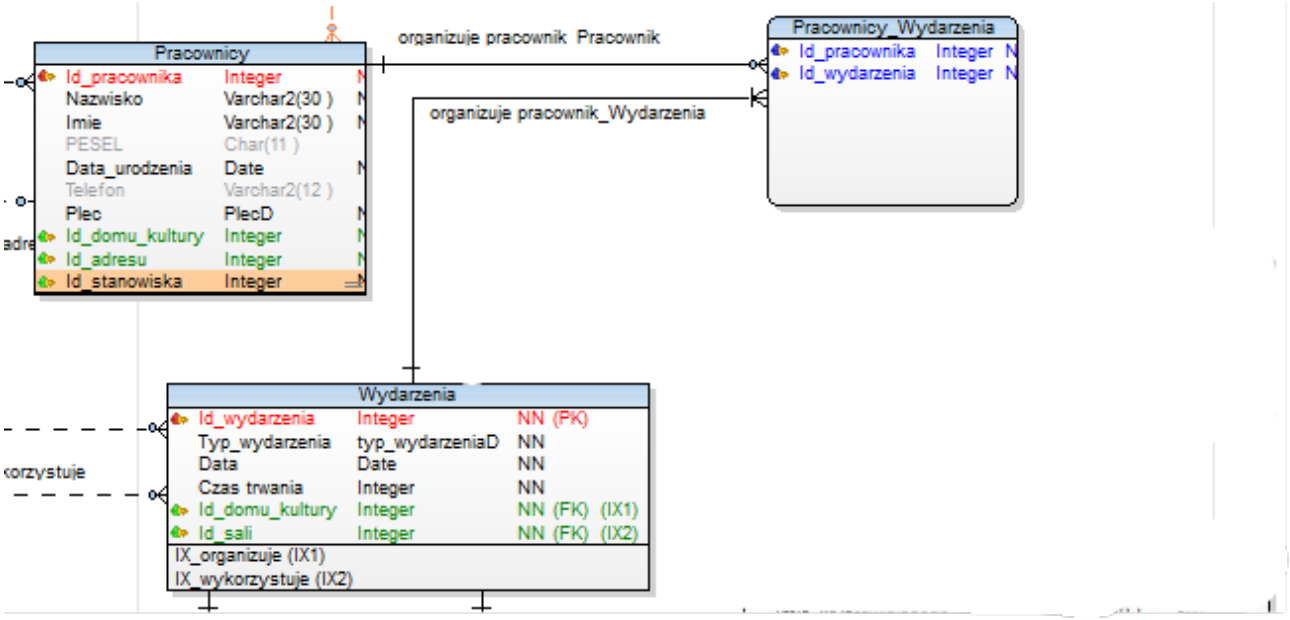
4.2 Usunięcie właściwości niekompatybilnych z modelem relacyjnych

Usunięcie niekompatybilnych właściwości polegało na utworzeniu przy relacjach wiele do wielu tablic łączących Przykłady: Zastąpienie relacji wiele do wielu pomiędzy encją Sale a zasoby używając tablicy łączącej

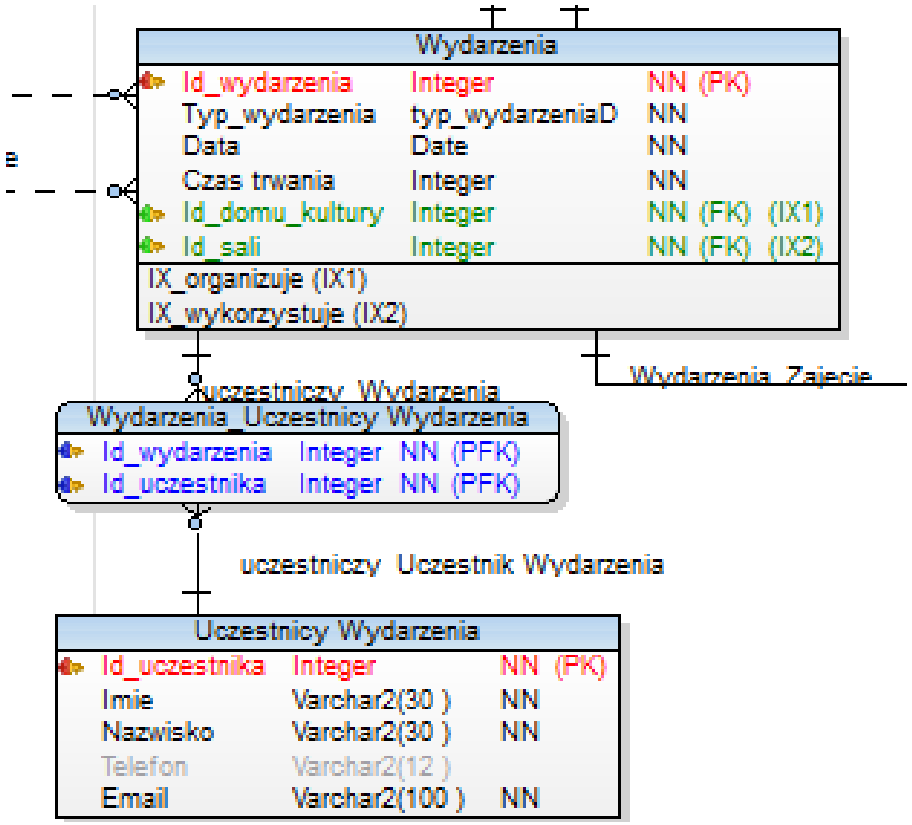


Rys. 2: Przykład usunięcia niekompatybilnych własności w modelu relacyjnym - relacja Sale-Zasoby

Kolejnymi podobnymi przykładami będzie zastąpienie takich samych relacji w przypadku Pracownicy, a Wydarzenia oraz Wydarzenia, a Uczestnicy wydarzenia



Rys. 3: Przykład usunięcia niekompatybilnych własności w modelu relacyjnym - relacja Pracownicy-Wydarzenia



Rys. 4: Przykład usunięcia niekompatybilnych własności w modelu relacyjnym - relacja Wydarzenia-Uczestnicy wydarzenia

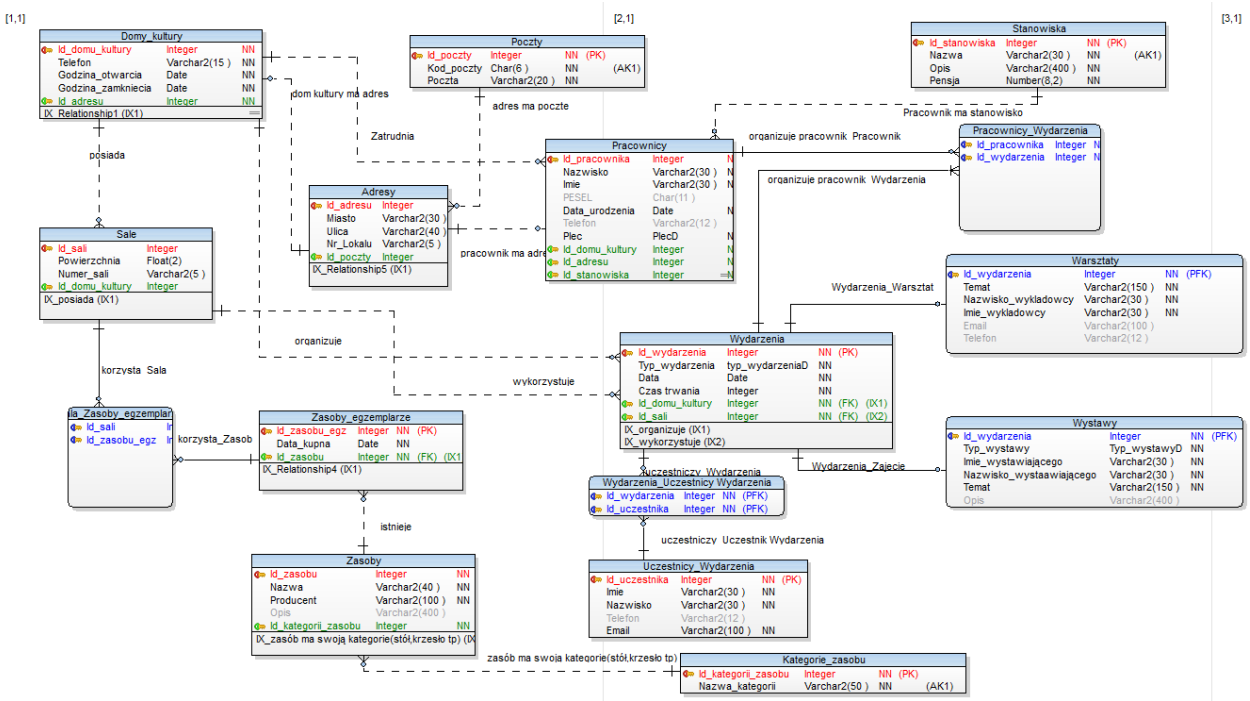
4.3 Proces normalizacji

Proces normalizacji polegał na doprowadzeniu naszego modelu logicznego do trzeciej postaci normalnej.

Żeby zapewnić trzecią postać normalną:

- Wyodrębniliśmy takie encję Adres, która jest połączona relacją z encją Domy_kultury oraz Wydarzenia (pozbyliśmy się redundancji danych)
- Stworzyliśmy encję Stanowiska, która zawiera wszystkie możliwe stanowiska w ośrodku i połączyliśmy ją relacją z encją Pracownicy
- Stworzyliśmy encje Poczty zawierające wszystkie używane kody pocztowe
- Utworzyliśmy encje Zasoby_egzemplarze które mają w sobie wszystkie zasoby używane w salach. Dodatkowo stworzyliśmy encje Zasoby która jest połaczona relacją z Zasoby_egzemplarze. Jest to tablica słownikowa, która zawiera w sobie informacje ogólne na temat jednego z zasobu. Chodzi o to że przykładowo możemy mieć kilka egzemplarzy np. stolika od tego samego procudenta o tej samej nazwie i aby uniknąć redundancji kodu (mając 100 tych samych stolików przy błędzie nazwy musielibysmy zmieniać w 100 miejscach a tak musimy tylko w jednym). Dodatkowo do encji zasoby połączyliśmy encje Kategorie_zasobów mówi o jaki ogólnie zasób chodzi czy o stolik czy o tablice etc.

4.4 Schemat ER na poziomie modelu relacyjnego



Rys. 5: Schemat modelu relacyjnego

4.5 Więzy integralności

- Domy_kultury
  - Id_domy_kultury - klucz głowny, not null
  - Telefon - nałożenie maski wprowadzania o formie, not null !000\-000\-000, not null
  - Godzina_otwarcia - not null

  - Godzina_zamkniecia - not null
  - Id_adresu - klucz obcy z encji Adresy, not null
- Sale
  - Id_sali - klucz głowny, not null
  - Powierzchnia - not null
  - Numer_sali - not null
  - Id_domy_kultury - klucz obcy z encji Domy_kultury, not null
- Adresy
  - Id_adresu - klucz głowny,not null
  - Miasto - not null
  - Ulica - not null
  - Nr_lokalu - not null
  - Id_poczty - klucz obcy z encji Poczty,not null
- Poczty
  - Id_poczty - klucz głowny, not null
  - Kod_poczty - nałożenie maski wprowadzania o formie, not null 00\-000
  - Poczta - not null, not null
- Pracownicy
  - Id_pracownika - klucz głowny, not null
  - Nazwisko - not null
  - Imie - not null
  - PESEL
  - Data_urodzenia - not null
  - Telefon - nałożenie maski wprowadzania o formie !000\-000\-000
  - Plec - wartości mogą być tylko z dziedziny {'K','M'}, not null
  - Id_domy_kultury - klucz obcy z encji Doomy_kultury, not null
  - Id_adresu - klucz obcy z encji Adresy, not null
  - Id_stanowiska - klucz obcy z encji Stanowiska, not null
- Stanowiska
  - Id_stanowiska - klucz głowny, not null
  - Nazwa - nazwa unikatowa, not null
  - Opis - not null
  - Pensja - not null
- Wydarzenia
  - Id_wydarzenia - klucz głowny, not null
  - Typ_wydarzenia - wartości mogą być tylko z dziedziny {'wystawa','warsztat'}, not null
  - Data - not null
  - Czas trwania - not null
  - Id_domu_kultury - klucz obcy z encji Domy_kultury, not null
  - Id_sali - klucz obcy z encji Sale, not null

- Uczestnicy_wydarzenia
  - Id_uczestnika -klucz główny, not null
  - Imie - not null
  - Nazwisko - not null
  - Telefon -
  - Email - not null
- Zasoby_egzemplarze
  - Id_zasobu_egz - klucz główny, not null
  - Data_kupna - not null
  - Id_zasobu - klucz obcy z encji Zasoby, not null
- Zasoby
  - Id_zasobu - klucz główny, not null
  - Nazwa - not null
  - Producent - not null
  - Opis -
  - Id_kategorii_zasobu - klucz obcy z encji Kategorie_zasobu, not null
- Kategorie_zasobu
  - Id_kategorii_zasobu - klucz główny, not null
  - Nazwa_kategorii - unikalna nazwa, not null

4.6 Proces denormalizacji

Uważamy że w przypadku naszego systemu denormalizacja nie jest potrzebna. Jest to stosunkowo mały system i nie musimy denormalizować w celu poprawy wydajności oraz szybkości naszego systemu.
Jednak, jeśli musielibyśmy zastosować denormalizację to uważamy że moglibyśmy zrezygnować z encji Kategorie_zasobu i jej relacji z encją Zasoby na rzecz dodania atrybutu 'kategoria' do encji Zasoby. Dziedziną tego atrybutu wtedy byłby zbiór np {"krzesło", "stolik", "biurko", "tablica", "laptop", "kreda", "płótna" }

5 Faza fizyczna

5.1 Projekt transakcji i weryfikacji ich wykonalności

- W obszarze obsługi pracowników są zdefiniowane działania dodawania oraz usuwania pracowników, modyfikacja danych osobowych, nadawanie stanowisk pracownikom oraz przypisanie pracownika do obsługi wydarzenia
- w obszarze domu kultury zostały zawarte modyfikowanie podstawowych wartości atrybutów
- w obszarze obsługi wydarzeń możemy wykonać operacje dodania wydarzenia oraz jego usuwania, podglądu zapisanych uczestników oraz modyfikacji informacji o wydarzeniu
- w obszarze uczestnika wydarzenia jesteśmy w stanie dodać i usunąć użytkownika z wydarzenia oraz sprawdzić w ilu wydarzeniach dany uczestnik uczestniczył
- w obszarze logistyki możemy dodawać oraz usuwać dostępne zasoby, dodawać poszczególne egzemplarze do sal oraz modyfikować informacje na temat przedmiotów

5.2 Strojenie bazy danych

Strojenie baz danych jest procesem polegającym na skróceniu czasu odpowiedzi serwera na przesyłane do niego zapytania. Jednym z etapów tego procesu jest indeksowanie. Indeksy są podstawowym mechanizmem, który służy do poprawy wydajności wykonywania zapytań w bazie danych. Jest to dodatkowa struktura danych, przechowywana obok tabeli, zoptymalizowana pod kątem wyszukiwania.

Korzystaliśmy się ze strategii "tylko indeks".

Strategia „tylko indeks" to metoda korzystając z której trzymamy się zasady, że wszystkie kolumny używane w zapytaniu są indeksowane. Dzięki temu serwer nie musi w ogóle sięgać do stron z danymi. Wszystkie dane zostaną odczytane z indeksów. Możemy dzięki temu osiągnąć dużą poprawę wydajności wykonywania zapytań. Zwykle jednak ze względu na dużą liczbę lub złożoność zakładanych indeksów istnieje ryzyko pogorszenia wydajności aktualizacji. Jednak wszystkie nasze klucze nie są złożone, więc nie musimy spotkać się z tym problemem.

5.3 Skrypt SQL zakładający bazę danych

```
/*
Created: 16.11.2020
Modified: 19.11.2020
Model: Dom Kultury
Database: Oracle 12c Release 2
*/


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

CREATE SEQUENCE PracownicySeq1
 INCREMENT BY 1
 START WITH 1
 NOMAXVALUE
 NOMINVALUE
 CACHE 20
/

CREATE SEQUENCE DomyKulturySeq1
 INCREMENT BY 1
 START WITH 1
 NOMAXVALUE
 NOMINVALUE
 CACHE 20
/

CREATE SEQUENCE PocztySeq1
 INCREMENT BY 1
 START WITH 1
 NOMAXVALUE
 NOMINVALUE
 CACHE 20
/

CREATE SEQUENCE AdresSeq1
 INCREMENT BY 1
 START WITH 1
```

```
 NOMAXVALUE
 NOMINVALUE
 CACHE 20
/

CREATE SEQUENCE SaleSeq1
 INCREMENT BY 1
 START WITH 1
 NOMAXVALUE
 NOMINVALUE
 CACHE 20
/

CREATE SEQUENCE ZasobyEgzSeq1
 INCREMENT BY 1
 START WITH 1
 NOMAXVALUE
 NOMINVALUE
 CACHE 20
/

CREATE SEQUENCE ZasobySeq1
 INCREMENT BY 1
 START WITH 1
 NOMAXVALUE
 NOMINVALUE
 CACHE 20
/

CREATE SEQUENCE WydarzeniaSeq1
 INCREMENT BY 1
 START WITH 1
 NOMAXVALUE
 NOMINVALUE
 CACHE 20
/

CREATE SEQUENCE KategorieZasobuSeq1
 INCREMENT BY 1
 START WITH 1
 NOMAXVALUE
 NOMINVALUE
 CACHE 20
/

CREATE SEQUENCE UczestnicyWydSeq1
 INCREMENT BY 1
 START WITH 1
 NOMAXVALUE
 NOMINVALUE
 CACHE 20
/

CREATE SEQUENCE StanowiskaSeq1
 INCREMENT BY 1
 START WITH 1
```

```
 NOMAXVALUE
 NOMINVALUE
 CACHE 20
/

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

-- Table Domy_kultury

CREATE TABLE Domy_kultury(
   Id_domu_kultury Integer NOT NULL,
   Telefon Varchar2(15 ) NOT NULL
         CONSTRAINT TelDomKult CHECK (Telefon LIKE '!000\-000\-000'),
   Godzina_otwarcia Date NOT NULL,
   Godzina_zamkniecia Date NOT NULL,
   Nr_adresu Integer NOT NULL
)
/

-- Create indexes for table Domy_kultury

CREATE INDEX IX_Relationship1 ON Domy_kultury (Nr_adresu)
/

-- Add keys for table Domy_kultury

ALTER TABLE Domy_kultury ADD CONSTRAINT Unique_Identifier1 PRIMARY KEY (
    Id_domu_kultury)
/

-- Table Pracownicy

CREATE TABLE Pracownicy(
   Id_pracownika Integer NOT NULL,
   Nazwisko Varchar2(30 ) NOT NULL,
   Imie Varchar2(30 ) NOT NULL,
   PESEL Char(11 ),
   Data_urodzenia Date NOT NULL,
   Telefon Varchar2(12 ),
   Plec Char(1 ) NOT NULL
         CHECK (Plec IN ('K', 'M')),
   Id_domu_kultury Integer NOT NULL,
   Nr_adresu Integer NOT NULL,
   Nr_stanowiska Integer NOT NULL
)
/

-- Create indexes for table Pracownicy

CREATE INDEX IX_Zatrudnia ON Pracownicy (Id_domu_kultury)
/

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

```
CREATE INDEX IX_Relationship3 ON Pracownicy (Nr_stanowiska)
/

-- Add keys for table Pracownicy

ALTER TABLE Pracownicy ADD CONSTRAINT Pracownik PK PRIMARY KEY (
    Id_pracownika)
/

-- Table Sale

CREATE TABLE Sale(
   Id_sali Integer NOT NULL,
   Powierzchnia Float(2) NOT NULL,
   Numer_sali Varchar2(5 ) NOT NULL,
   Id_domu_kultury Integer NOT NULL
)
/

-- Create indexes for table Sale

CREATE INDEX IX_posiada ON Sale (Id_domu_kultury)
/

-- Add keys for table Sale

ALTER TABLE Sale ADD CONSTRAINT Unique_Identifier2 PRIMARY KEY (Id_sali)
/

-- Table Zasoby_egzemplarze

CREATE TABLE Zasoby_egzemplarze(
   Id_zasobu_egz Integer NOT NULL,
   Data_kupna Date NOT NULL,
   Id_zasobu Integer NOT NULL
)
/

-- Create indexes for table Zasoby_egzemplarze

CREATE INDEX IX_Relationship4 ON Zasoby_egzemplarze (Id_zasobu)
/

-- Add keys for table Zasoby_egzemplarze

ALTER TABLE Zasoby_egzemplarze ADD CONSTRAINT Unique_Identifier4 PRIMARY
    KEY (Id_zasobu_egz)
/

-- Table Wydarzenia

CREATE TABLE Wydarzenia(
   Id_wydarzenia Integer NOT NULL,
   Typ_wydarzenia Varchar2(30 ) NOT NULL
         CHECK (Typ_wydarzenia IN ('wystawa','warsztat')),
   Data_wydarzenia Date NOT NULL,
```

```
    Czas_trwania Integer NOT NULL,
    Id_domu_kultury Integer NOT NULL,
    Id_sali Integer NOT NULL
)
/

-- Create indexes for table Wydarzenia

CREATE INDEX IX_organizuje ON Wydarzenia (Id_domu_kultury)
/

CREATE INDEX IX_wykorzystuje ON Wydarzenia (Id_sali)
/

-- Add keys for table Wydarzenia

ALTER TABLE Wydarzenia ADD CONSTRAINT Unique_Identifier5 PRIMARY KEY (
    Id_wydarzenia)
/

-- Table Uczestnicy_Wydarzenia

CREATE TABLE Uczestnicy_Wydarzenia(
    Id_uczestnika Integer NOT NULL,
    Imie Varchar2(30 ) NOT NULL,
    Nazwisko Varchar2(30 ) NOT NULL,
    Telefon Varchar2(12 ),
    Email Varchar2(100 ) NOT NULL
)
/

-- Add keys for table Uczestnicy_Wydarzenia

ALTER TABLE Uczestnicy_Wydarzenia ADD CONSTRAINT Unique_Identifier6
    PRIMARY KEY (Id_uczestnika)
/

-- Table Warsztaty

CREATE TABLE Warsztaty(
    Id_wydarzenia Integer NOT NULL,
    Temat Varchar2(150 ) NOT NULL,
    Nazwisko_wykladowcy Varchar2(30 ) NOT NULL,
    Imie_wykladowcy Varchar2(30 ) NOT NULL,
    Email Varchar2(100 ),
    Telefon Varchar2(12 )
)
/

-- Add keys for table Warsztaty

ALTER TABLE Warsztaty ADD CONSTRAINT Unique_Identifier7 PRIMARY KEY (
    Id_wydarzenia)
/

-- Table Wystawy
```

```
CREATE TABLE Wystawy(
   Id_wydarzenia Integer NOT NULL,
   Typ_wystawy Varchar2(50 ) NOT NULL
         CHECK (Typ_wystawy IN ('malarska','fotograficzna','interaktywna',
              'muzyczna','filmowa')),
   Imie_wystawiającego Varchar2(30 ) NOT NULL,
   Nazwisko_wystaawiającego Varchar2(30 ) NOT NULL,
   Temat Varchar2(150 ) NOT NULL,
   Opis Varchar2(400 )
)
/

-- Add keys for table Wystawy

ALTER TABLE Wystawy ADD CONSTRAINT Unique_Identifier8 PRIMARY KEY (
    Id_wydarzenia)
/

-- Table Sala_Zasoby_egzemplarze

CREATE TABLE Sala_Zasoby_egzemplarze(
   Id_sali Integer NOT NULL,
   Id_zasobu Integer NOT NULL
)
/

-- Table Pracownicy_Wydarzenia

CREATE TABLE Pracownicy_Wydarzenia(
   Id_pracownika Integer NOT NULL,
   Id_wydarzenia Integer NOT NULL
)
/

-- Table Wydarzenia_Uczestnicy Wydarzenia

CREATE TABLE Wydarzenia_Uczestnicy Wydarzenia(
   Id_wydarzenia Integer NOT NULL,
   Id_uczestnika Integer NOT NULL
)
/

-- Table Adresy

CREATE TABLE Adresy(
   Id_adresu Integer NOT NULL,
   Miasto Varchar2(30 ) NOT NULL,
   Ulica Varchar2(40 ) NOT NULL,
   Nr_Lokalu Varchar2(5 ) 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 (Id_adresu)
/

ALTER TABLE Adresy ADD CONSTRAINT Nr_adresu UNIQUE (Id_adresu)
/

-- Table Stanowiska

CREATE TABLE Stanowiska(
    Id_stanowiska Integer NOT NULL,
    Nazwa Varchar2(30 ) NOT NULL,
    Opis Varchar2(400 ) NOT NULL,
    Pensja Number(8,2) 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 Zasoby

CREATE TABLE Zasoby(
    Id_zasobu Integer NOT NULL,
    Nazwa Varchar2(40 ) NOT NULL,
    Producent Varchar2(100 ) NOT NULL,
    Opis Varchar2(400 ),
    Id_kategorii_zasobu Integer NOT NULL
)
/

-- Create indexes for table Zasoby

CREATE INDEX IX_zasób ma swoją kategorie(stół,krzesło tp) ON Zasoby (
    Id_kategorii_zasobu)
/

-- Add keys for table Zasoby

ALTER TABLE Zasoby ADD CONSTRAINT PK_Zasoby PRIMARY KEY (Id_zasobu)
/

ALTER TABLE Zasoby ADD CONSTRAINT Id_zasobu UNIQUE (Id_zasobu)
/

-- Table Poczty
```

```
CREATE TABLE Poczty(
   Id_poczty Integer NOT NULL,
   Kod_poczty Char(6 ) NOT NULL
         CONSTRAINT KodPocztowy CHECK (Kod_Poczty LIKE '00\-000'),
   Poczta Varchar2(20 ) NOT NULL
)
/

-- Add keys for table Poczty

ALTER TABLE Poczty ADD CONSTRAINT PK_Poczty PRIMARY KEY (Id_poczty)
/

ALTER TABLE Poczty ADD CONSTRAINT Kod_poczty UNIQUE (Kod_poczty)
/

-- Table Kategorie_zasobu

CREATE TABLE Kategorie_zasobu(
   Id_kategorii_zasobu Integer NOT NULL,
   Nazwa_kategorii Varchar2(50 ) NOT NULL
)
/

-- Add keys for table Kategorie_zasobu

ALTER TABLE Kategorie_zasobu ADD CONSTRAINT PK_Kategorie_zasobu PRIMARY
    KEY (Id_kategorii_zasobu)
/

ALTER TABLE Kategorie_zasobu ADD CONSTRAINT Nazwa_kategorii UNIQUE (
    Nazwa_kategorii)
/

-- Trigger for sequence DomyKulturySeq1 for column Id_domu_kultury in
    table Domy_kultury ----------
CREATE OR REPLACE TRIGGER ts_Domy_kultury_DomyKulturySeq1 BEFORE INSERT
ON Domy_kultury FOR EACH ROW
BEGIN
   :new.Id_domu_kultury := DomyKulturySeq1.nextval;
END;
/
CREATE OR REPLACE TRIGGER tsu_Domy_kultury_DomyKulturySeq1 AFTER UPDATE
    OF Id_domu_kultury
ON Domy_kultury FOR EACH ROW
BEGIN
   RAISE_APPLICATION_ERROR(-20010,'Cannot update column Id_domu_kultury in
       table Domy_kultury as it uses sequence.');
END;
/

-- Trigger for sequence PracownicySeq1 for column Id_pracownika in table
    Pracownicy ----------
CREATE OR REPLACE TRIGGER ts_Pracownicy_PracownicySeq1 BEFORE INSERT
ON Pracownicy FOR EACH ROW
```

```
BEGIN
   :new.Id_pracownika := PracownicySeq1.nextval;
END;
/
CREATE OR REPLACE TRIGGER tsu_Pracownicy_PracownicySeq1 AFTER UPDATE OF
    Id_pracownika
ON Pracownicy FOR EACH ROW
BEGIN
   RAISE_APPLICATION_ERROR(-20010,'Cannot update column Id_pracownika in
       table Pracownicy as it uses sequence.');
END;
/

-- Trigger for sequence SaleSeq1 for column Id_sali in table Sale
    ----------
CREATE OR REPLACE TRIGGER ts_Sale_SaleSeq1 BEFORE INSERT
ON Sale FOR EACH ROW
BEGIN
   :new.Id_sali := SaleSeq1.nextval;
END;
/
CREATE OR REPLACE TRIGGER tsu_Sale_SaleSeq1 AFTER UPDATE OF Id_sali
ON Sale FOR EACH ROW
BEGIN
   RAISE_APPLICATION_ERROR(-20010,'Cannot update column Id_sali in table
       Sale as it uses sequence.');
END;
/

-- Trigger for sequence ZasobyEgzSeq1 for column Id_zasobu_egz in table
    Zasoby_egzemplarze ----------
CREATE OR REPLACE TRIGGER ts_Zasoby_egzemplarze_ZasobyEgzSeq1 BEFORE
    INSERT
ON Zasoby_egzemplarze FOR EACH ROW
BEGIN
   :new.Id_zasobu_egz := ZasobyEgzSeq1.nextval;
END;
/
CREATE OR REPLACE TRIGGER tsu_Zasoby_egzemplarze_ZasobyEgzSeq1 AFTER
    UPDATE OF Id_zasobu_egz
ON Zasoby_egzemplarze FOR EACH ROW
BEGIN
   RAISE_APPLICATION_ERROR(-20010,'Cannot update column Id_zasobu_egz in
       table Zasoby_egzemplarze as it uses sequence.');
END;
/

-- Trigger for sequence WydarzeniaSeq1 for column Id_wydarzenia in table
    Wydarzenia ----------
CREATE OR REPLACE TRIGGER ts_Wydarzenia_WydarzeniaSeq1 BEFORE INSERT
ON Wydarzenia FOR EACH ROW
BEGIN
   :new.Id_wydarzenia := WydarzeniaSeq1.nextval;
END;
/
```

```
CREATE OR REPLACE TRIGGER tsu_Wydarzenia_WydarzeniaSeq1 AFTER UPDATE OF
    Id_wydarzenia
ON Wydarzenia FOR EACH ROW
BEGIN
  RAISE_APPLICATION_ERROR(-20010,'Cannot update column Id_wydarzenia in
      table Wydarzenia as it uses sequence.');
END;
/

-- Trigger for sequence UczestnicyWydSeq1 for column Id_uczestnika in
    table Uczestnicy_Wydarzenia ----------
CREATE OR REPLACE TRIGGER ts_Uczestnicy_Wydarzenia_UczestnicyWydSeq1
    BEFORE INSERT
ON Uczestnicy_Wydarzenia FOR EACH ROW
BEGIN
  :new.Id_uczestnika := UczestnicyWydSeq1.nextval;
END;
/
CREATE OR REPLACE TRIGGER tsu_Uczestnicy_Wydarzenia_UczestnicyWydSeq1
    AFTER UPDATE OF Id_uczestnika
ON Uczestnicy_Wydarzenia FOR EACH ROW
BEGIN
  RAISE_APPLICATION_ERROR(-20010,'Cannot update column Id_uczestnika in
      table Uczestnicy_Wydarzenia as it uses sequence.');
END;
/

-- Trigger for sequence AdresSeq1 for column Id_adresu in table Adresy
    ----------
CREATE OR REPLACE TRIGGER ts_Adresy_AdresSeq1 BEFORE INSERT
ON Adresy FOR EACH ROW
BEGIN
  :new.Id_adresu := AdresSeq1.nextval;
END;
/
CREATE OR REPLACE TRIGGER tsu_Adresy_AdresSeq1 AFTER UPDATE OF Id_adresu
ON Adresy FOR EACH ROW
BEGIN
  RAISE_APPLICATION_ERROR(-20010,'Cannot update column Id_adresu in table
      Adresy as it uses sequence.');
END;
/

-- Trigger for sequence StanowiskaSeq1 for column Id_stanowiska in table
    Stanowiska ----------
CREATE OR REPLACE TRIGGER ts_Stanowiska_StanowiskaSeq1 BEFORE INSERT
ON Stanowiska FOR EACH ROW
BEGIN
  :new.Id_stanowiska := StanowiskaSeq1.nextval;
END;
/
CREATE OR REPLACE TRIGGER tsu_Stanowiska_StanowiskaSeq1 AFTER UPDATE OF
    Id_stanowiska
ON Stanowiska FOR EACH ROW
BEGIN
```

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

-- Trigger for sequence ZasobySeq1 for column Id_zasobu in table Zasoby
    ----------
CREATE OR REPLACE TRIGGER ts_Zasoby_ZasobySeq1 BEFORE INSERT
ON Zasoby FOR EACH ROW
BEGIN
  :new.Id_zasobu := ZasobySeq1.nextval;
END;
/
CREATE OR REPLACE TRIGGER tsu_Zasoby_ZasobySeq1 AFTER UPDATE OF Id_zasobu
ON Zasoby FOR EACH ROW
BEGIN
  RAISE_APPLICATION_ERROR(-20010,'Cannot update column Id_zasobu in table
      Zasoby as it uses sequence.');
END;
/

-- Trigger for sequence PocztySeq1 for column Id_poczty in table Poczty
    ----------
CREATE OR REPLACE TRIGGER ts_Poczty_PocztySeq1 BEFORE INSERT
ON Poczty FOR EACH ROW
BEGIN
  :new.Id_poczty := PocztySeq1.nextval;
END;
/
CREATE OR REPLACE TRIGGER tsu_Poczty_PocztySeq1 AFTER UPDATE OF Id_poczty
ON Poczty FOR EACH ROW
BEGIN
  RAISE_APPLICATION_ERROR(-20010,'Cannot update column Id_poczty in table
      Poczty as it uses sequence.');
END;
/

-- Trigger for sequence KategorieZasobuSeq1 for column
    Id_kategorii_zasobu in table Kategorie_zasobu ----------
CREATE OR REPLACE TRIGGER ts_Kategorie_zasobu_KategorieZasobuSeq1 BEFORE
    INSERT
ON Kategorie_zasobu FOR EACH ROW
BEGIN
  :new.Id_kategorii_zasobu := KategorieZasobuSeq1.nextval;
END;
/
CREATE OR REPLACE TRIGGER tsu_Kategorie_zasobu_KategorieZasobuSeq1 AFTER
    UPDATE OF Id_kategorii_zasobu
ON Kategorie_zasobu FOR EACH ROW
BEGIN
  RAISE_APPLICATION_ERROR(-20010,'Cannot update column
      Id_kategorii_zasobu in table Kategorie_zasobu as it uses sequence.')
      ;
END;
/
```

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

ALTER TABLE Wydarzenia ADD CONSTRAINT organizuje FOREIGN KEY (
    Id_domu_kultury) REFERENCES Domy_kultury (Id_domu_kultury)
/


ALTER TABLE Wydarzenia ADD CONSTRAINT wykorzystuje FOREIGN KEY (Id_sali)
    REFERENCES Sale (Id_sali)
/


ALTER TABLE Sale ADD CONSTRAINT posiada FOREIGN KEY (Id_domu_kultury)
    REFERENCES Domy_kultury (Id_domu_kultury)
/


ALTER TABLE Pracownicy ADD CONSTRAINT Zatrudnia FOREIGN KEY (
    Id_domu_kultury) REFERENCES Domy_kultury (Id_domu_kultury)
/


ALTER TABLE Domy_kultury ADD CONSTRAINT dom kultury ma adres FOREIGN KEY
    (Nr_adresu) REFERENCES Adresy (Id_adresu)
/


ALTER TABLE Pracownicy ADD CONSTRAINT pracownik ma adres FOREIGN KEY (
    Nr_adresu) REFERENCES Adresy (Id_adresu)
/


ALTER TABLE Pracownicy ADD CONSTRAINT Pracownik ma stanowisko FOREIGN KEY
     (Nr_stanowiska) REFERENCES Stanowiska (Id_stanowiska)
/


ALTER TABLE Zasoby_egzemplarze ADD CONSTRAINT istnieje FOREIGN KEY (
    Id_zasobu) REFERENCES Zasoby (Id_zasobu)
/


ALTER TABLE Adresy ADD CONSTRAINT adres ma poczte FOREIGN KEY (Nr_poczty)
     REFERENCES Poczty (Id_poczty)
/
```

```
ALTER TABLE Zasoby ADD CONSTRAINT zasób ma swoją kategorie(stół,krzesło
    tp) FOREIGN KEY (Id_kategorii_zasobu) REFERENCES Kategorie_zasobu (
    Id_kategorii_zasobu)
/
```

5.4 Przykłady zapytań i poleceń SQL odnoszących się do bazy danych

Wypełniliśmy danymi dwie tabele: Kategorie_zasobu oraz Zasoby. Przykłady wprowadzenie jednego rekordu jest poniżej. Dla sprawdzenia czy wszystko działa jak powinno wykonaliśmy polecenie SELECT

```
INSERT INTO Kategorie_zasobu VALUES(KATEGORIEZASOBUSEQ1.nextval, 'biurko'
    )

INSERT INTO Zasoby VALUES(ZASOBYSEQ1.nextval, 'FLEXTAPE','FLEX', NULL, 2)
    ;

SELECT zasoby.nazwa, zasoby.producent,kategorie_zasobu.nazwa_kategorii,
    zasoby.opis from Zasoby INNER JOIN Kategorie_zasobu ON ZASOBY.
    Id_kategorii_zasobu=kategorie_zasobu.id_kategorii_zasobu
```

	NAZWA	PRODUCENT	NAZWA_KATEGORII	OPIS
1	FLUX	IKEA	krzeslo	(null)
2	ALOE	Biedronka	biurko	(null)
3	ALOEs	Biedronka	biurko	(null)
4	ALCA	Biedronka	stolik	(null)
5	FLEXTAPE	FLEX	krzeslo	(null)

Rys. 6: Działajace polecenie SELECT wraz z INNER JOIN

Jak się spodziewaliśmy otrzymaliśmy poprawnie stworzone relacje, które działają i zachowują się w sposób prawidłowy.