

# **Podstawy Systemów Baz Danych**

Dokumentacja projektu

**“Wigilia dla ubogich”**

Wykonali: Piotr Pokorski - analityk  
Arkadiusz Piórkowski - projektant  
Tomasz Firynowicz - programista  
Marcin Zajęczkowski - programista  
Grupa 47

# **Spis treści**

<b>Harmonogram</b>	<b>2</b>
<b>Notatka</b>	<b>2</b>
<b>Aktorzy</b>	<b>3</b>
<b>Przypadki użycia</b>	<b>4</b>
<b>Diagram czynności</b>	<b>7</b>
<b>Model EER</b>	<b>17</b>
<b>Model relacyjny</b>	<b>18</b>
<b>Projekt GUI</b>	<b>19</b>
<b>Projekty raportów</b>	<b>20</b>
Statystyki darczyńców	20
Statystyki stołów	20
<b>Implementacja bazy danych</b>	<b>20</b>
<b>Implementacja GUI</b>	<b>20</b>
Zakładka “Goście”	21
Okno dodawania gościa	22
Okno przypisania do stołu	22
Okno edycji preferencji	22
Okno “Wyszukaj gościa”	23
Okno “Edycja gościa”	23
Okno “Edycja imienia lub nazwiska”	23
Okno “Edycja preferencji”	24
Okno “Zmień numer stolika”	24
Okno “Edytuj prezenty”	24
Raport “Darczyńcy prawni”	25
Raport “Darczyńcy fizyczni”	26
Raport “Statystyki stołów”	26

## **1. Harmonogram**

<b>Data</b>	<b>Zadanie</b>
07.11.2016	opis procesu w postaci notatki - specyfikacja wymagań
07.11.2016-21.11.2016	model EER - 1.etap
21.11.2016-12.12.2016	przypadki użycia -1. etap; model relacyjny - 2.etap
12.12.2016-23.12.2016	diagram czynności -1. etap; szczegółowy projekt GUI oraz przejść między ekranami, opracowanie raportów, - 2. etap
23.12.2016-30.01.2017	Implementacja bazy danych, wprowadzenie przykładowych danych, implementacja GUI, implementacja raportów-3. etap

## **2. Notatka**

Celem projektu jest zaprojektowanie bazy danych do obsługi przygotowania wydarzenia "Wigilia z ubogimi" organizowanego przez wspólnotę Sant'Egidio w Warszawie. Wydarzenie organizowane jest co roku w jednym miejscu w Warszawie.

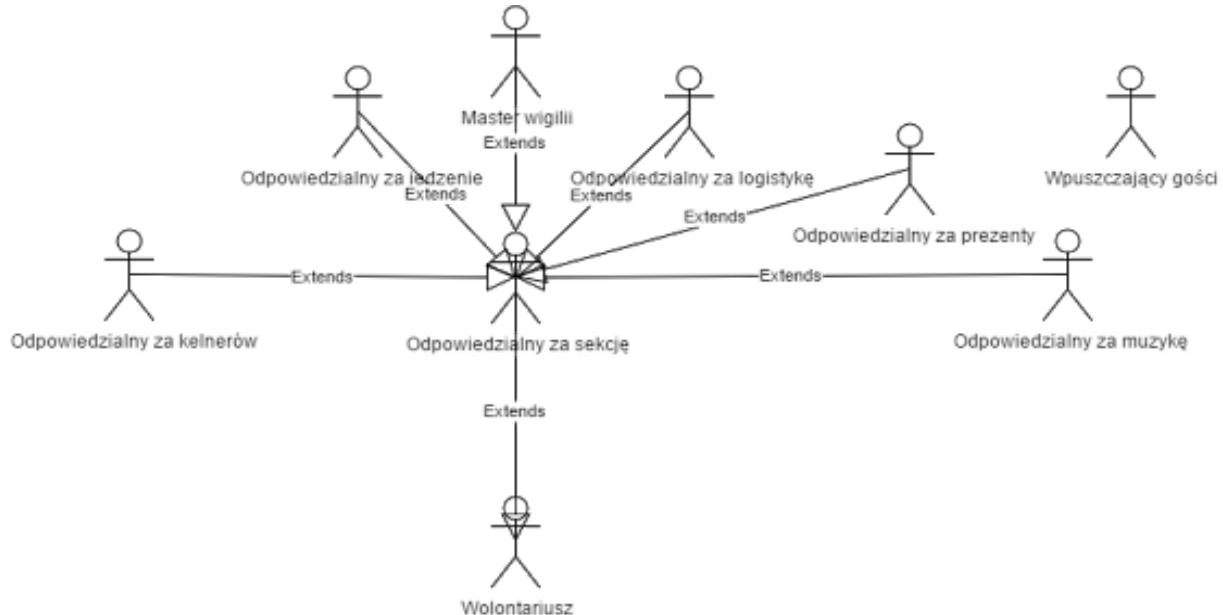
Organizacja wigilii dzieli się na kilka sekcji organizacyjnych:

1. Logistyka - jej celem jest głównie transport rzeczy: darów (jedzenia i prezentów) oraz sprzętu. Każdą rzeczą należy zabrać z określonego adresu w pewnym przedziale czasu, należy ją też przetransportować pod odpowiedni adres. Każdy taki transport ma kontakt do darczyńcy i ma przydzieloną osobę, która transport realizuje. Sekcja może organizować sprzęt.
2. Dekoracje - jej celem jest przystrojenie sali na wigilię.
3. Jedzenie - jej celem jest pozyskanie jedzenia od sklepów/restauracji/osób prywatnych. Następnie te dane potrzebne są logistyczce. Sekcja może organizować dar - jedzenie.
4. Prezenty - jej celem jest pozyskanie rzeczy, z których potem zostaną złożone prezenty - każdy gość otrzymuje jakiś prezent. Sekcja może organizować dar - prezent.
5. Muzyka - jej celem jest granie i śpiewanie. Członkowie chóru, każdy śpiewa lub gra na jakimś instrumencie. Sekcja może organizować sprzęt.

Każda sekcja ma 3 osoby odpowiedzialne. Każda z nich ma możliwość dodawania gości oraz dodawania wolontariuszy, oraz podgląd list swoich sekcji.

Grupa jedzeniowa i prezentowa mogą przyjmować dary zarówno od osób prywatnych, jak i sklepów, restauracji, itd.

### 3. Aktorzy



#### a. Odpowiedzialny za sekcję

- zarządza wolontariuszami - ma możliwość dodawania, edycji, przypisywania do sekcji
- ma możliwość dodawania i edycji darczyńców
- generuje raporty - wolontariuszy oraz przedmiotów
  - i. **Odpowiedzialny za logistykę**
    - dodaje i organizuje stoły
    - dodaje i edytuje wszystkie rodzaje przedmiotów
    - organizuje transporty
  - ii. **Odpowiedzialny za prezenty**
    - organizuje prezenty i dodaje je do bazy
  - iii. **Odpowiedzialny za jedzenie**
    - organizuje jedzenie
  - iv. **Odpowiedzialny za muzykę**
    - dodaje muzyków
    - organizuje sprzęt
  - v. **Odpowiedzialny za kelnerów**
    - dodaje kelnerów
    - przypisuje kelnerów do stołów, zaznacza ich obecność na odprawie

#### b. Master wigilli

- zakłada nową wigilię
- dodaje i przypisuje gości do stołów

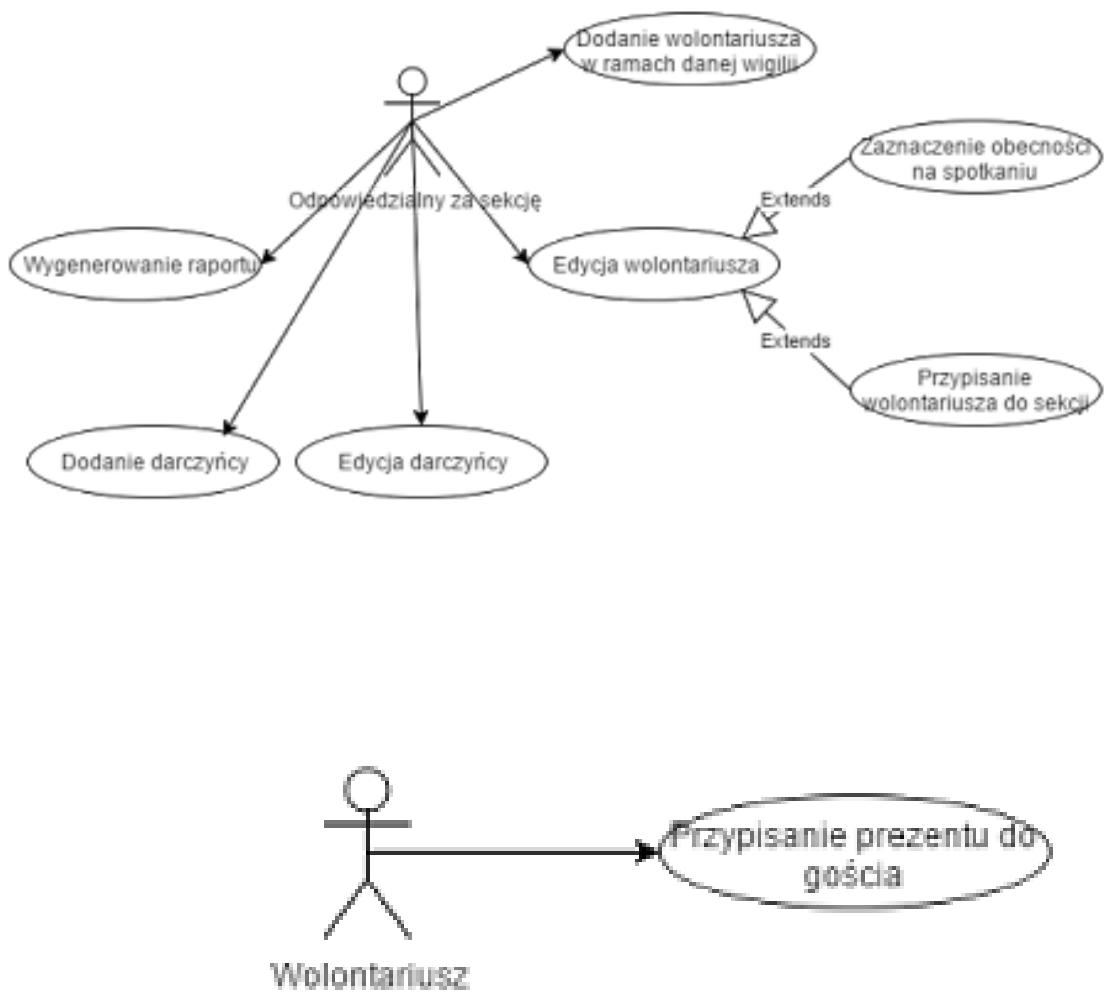
#### c. Wolontariusz

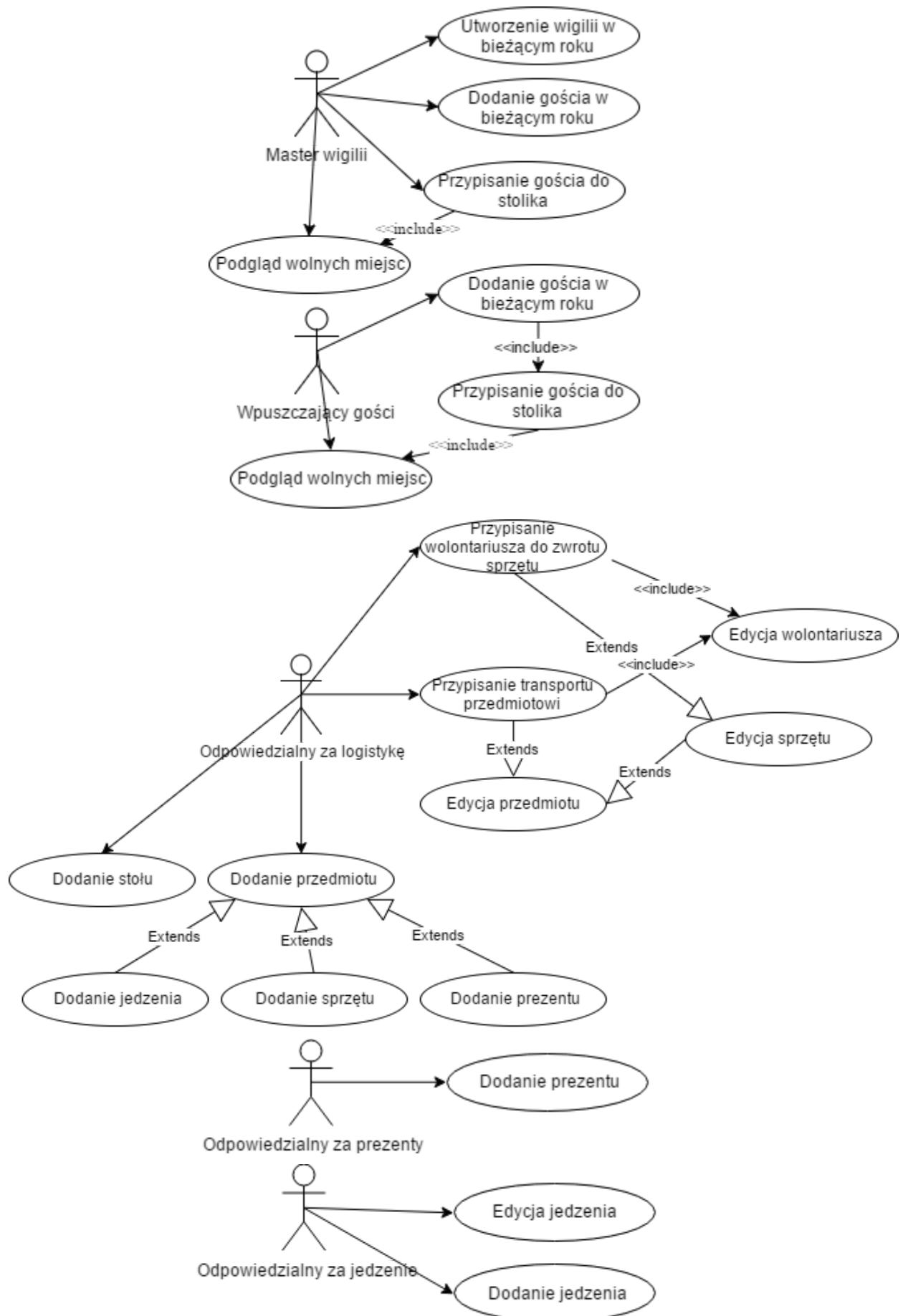
- przypisuje gościom prezenty

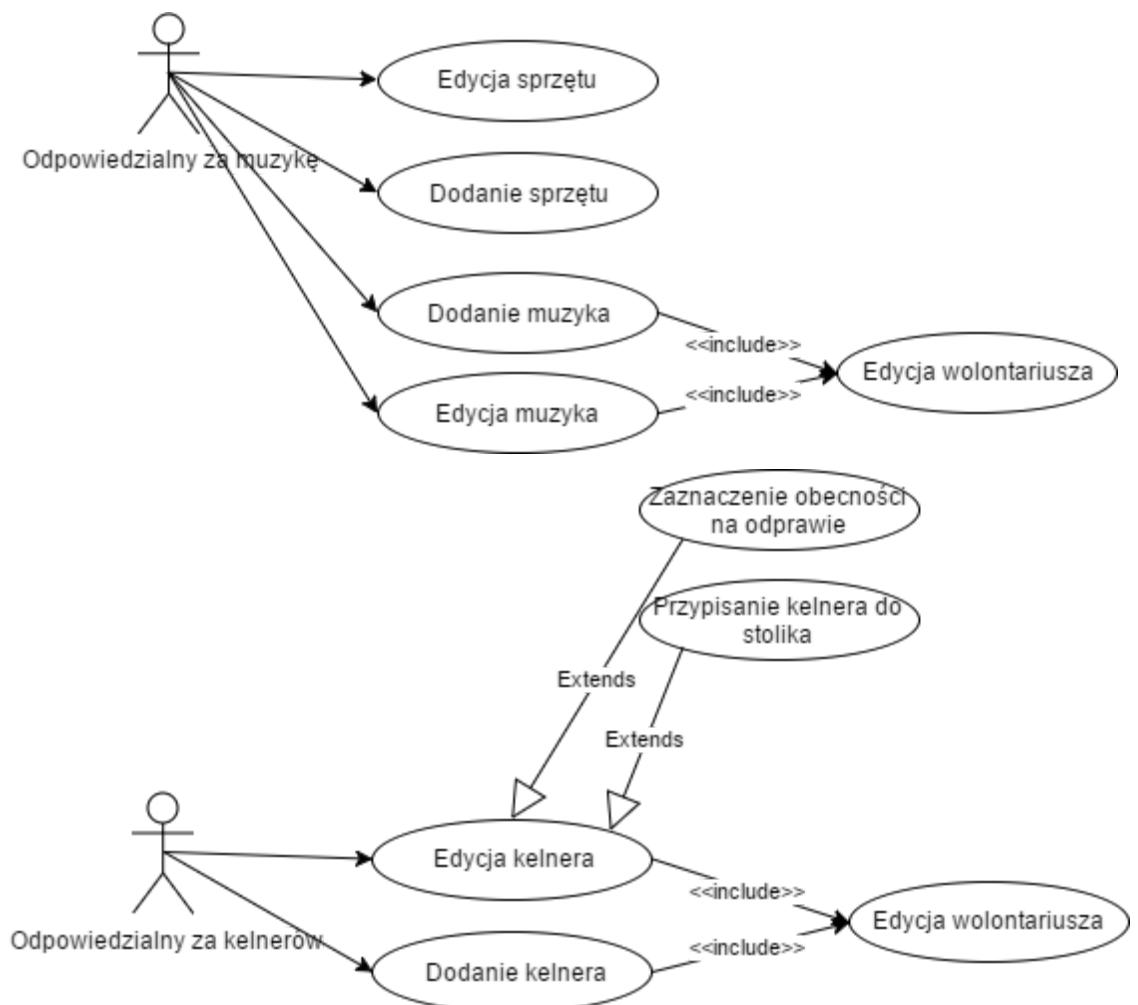
#### d. Wpuszczający gości

- dodaje gości, musi ich jednocześnie przypisać do stołu

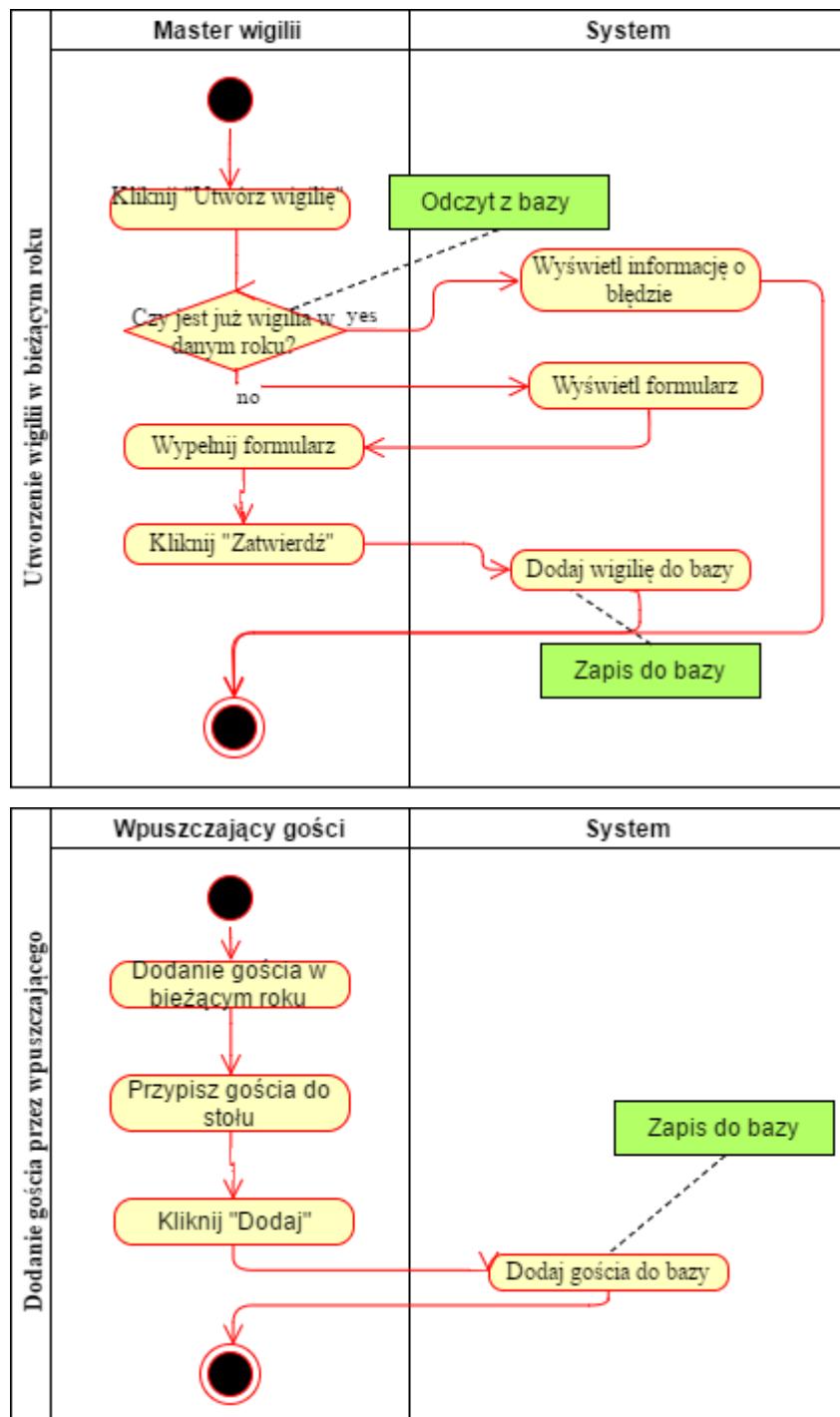
## 4. Przypadki użycia

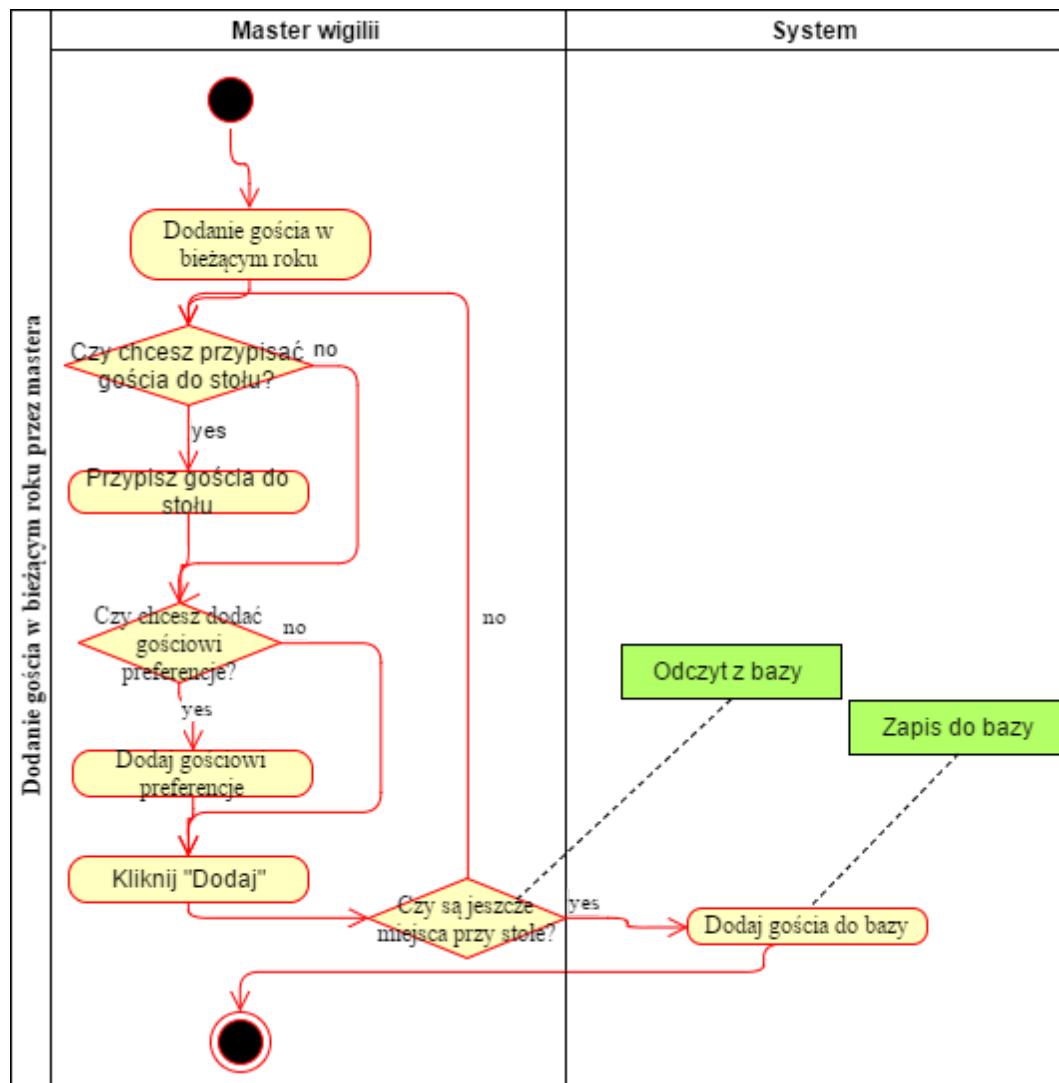


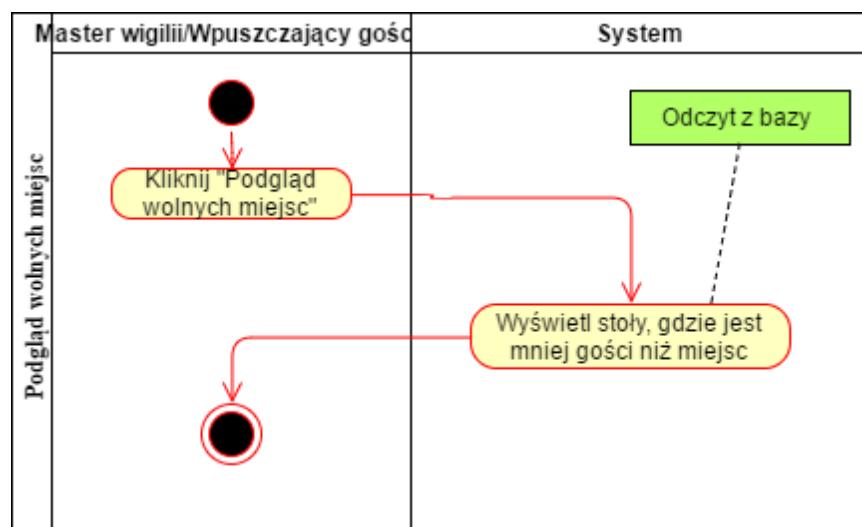
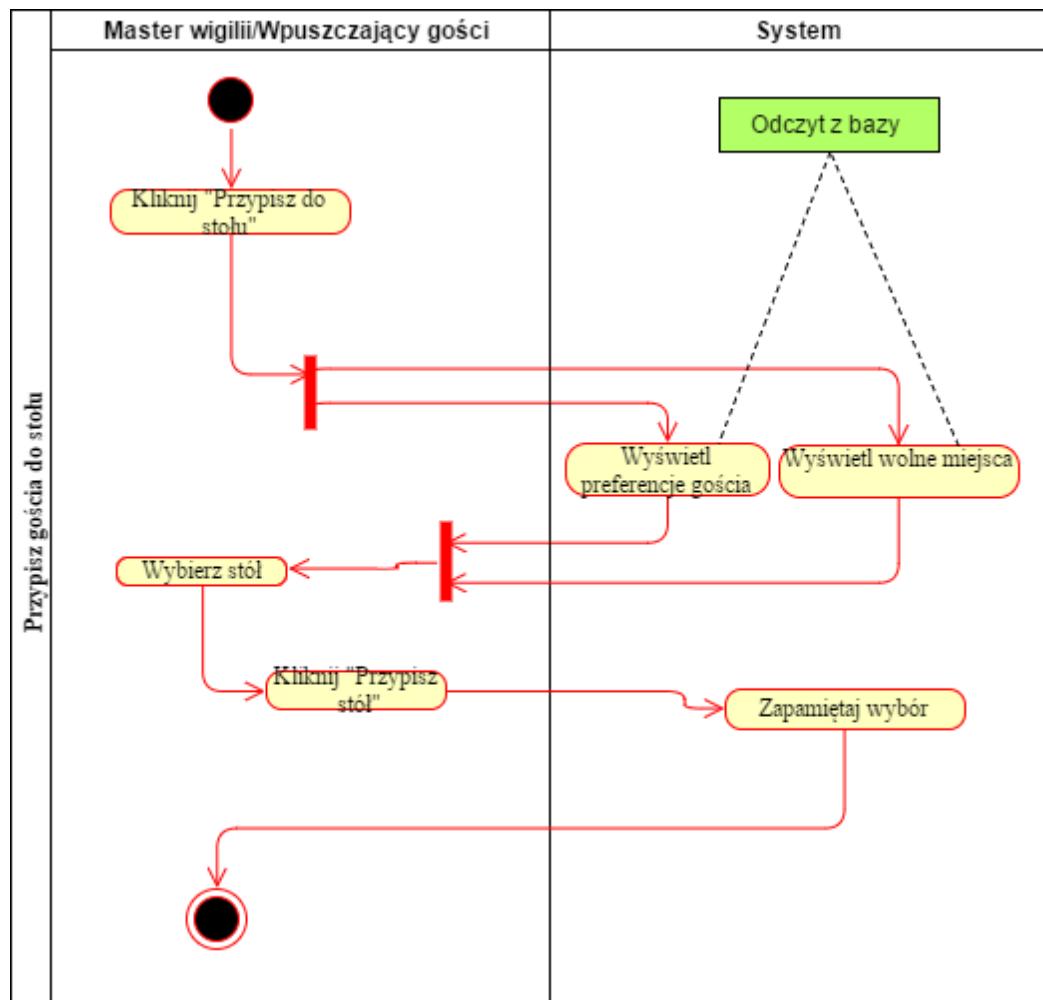


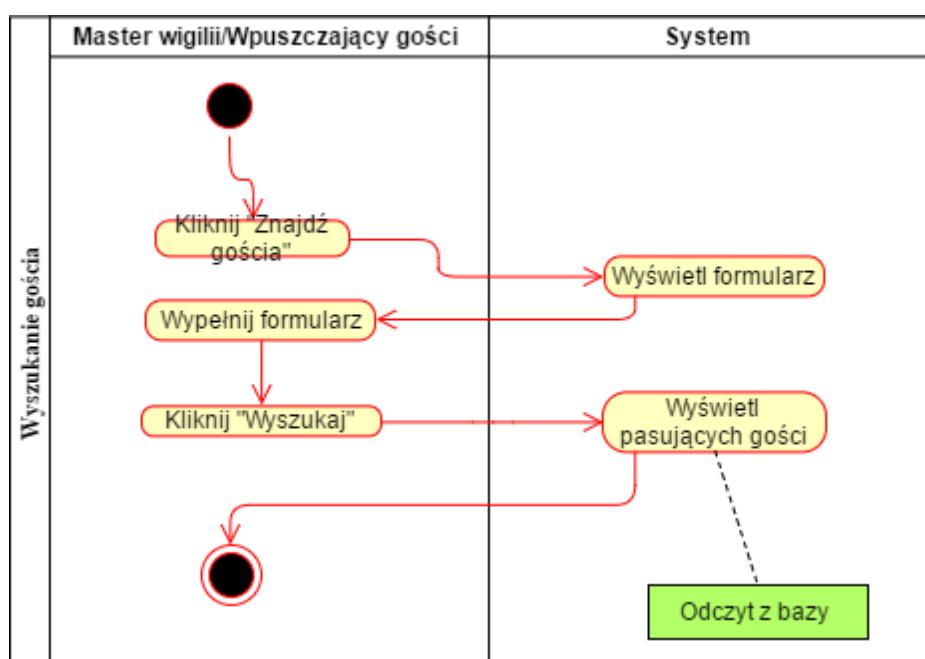
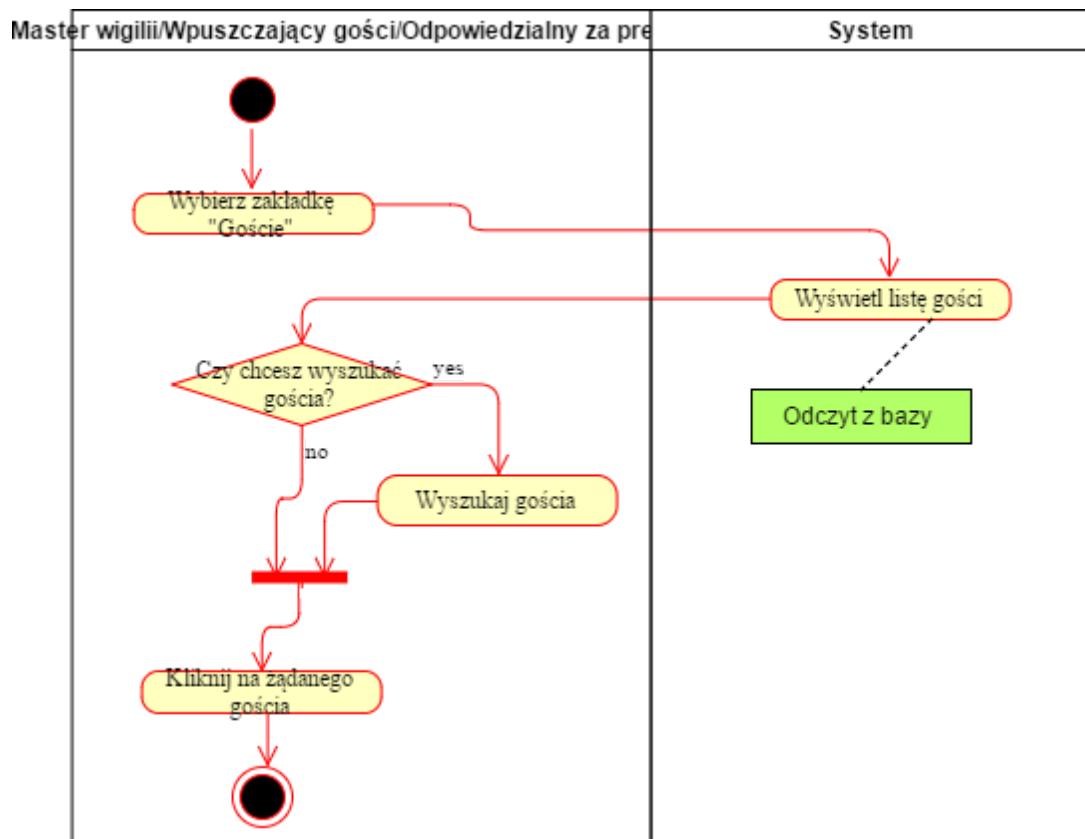


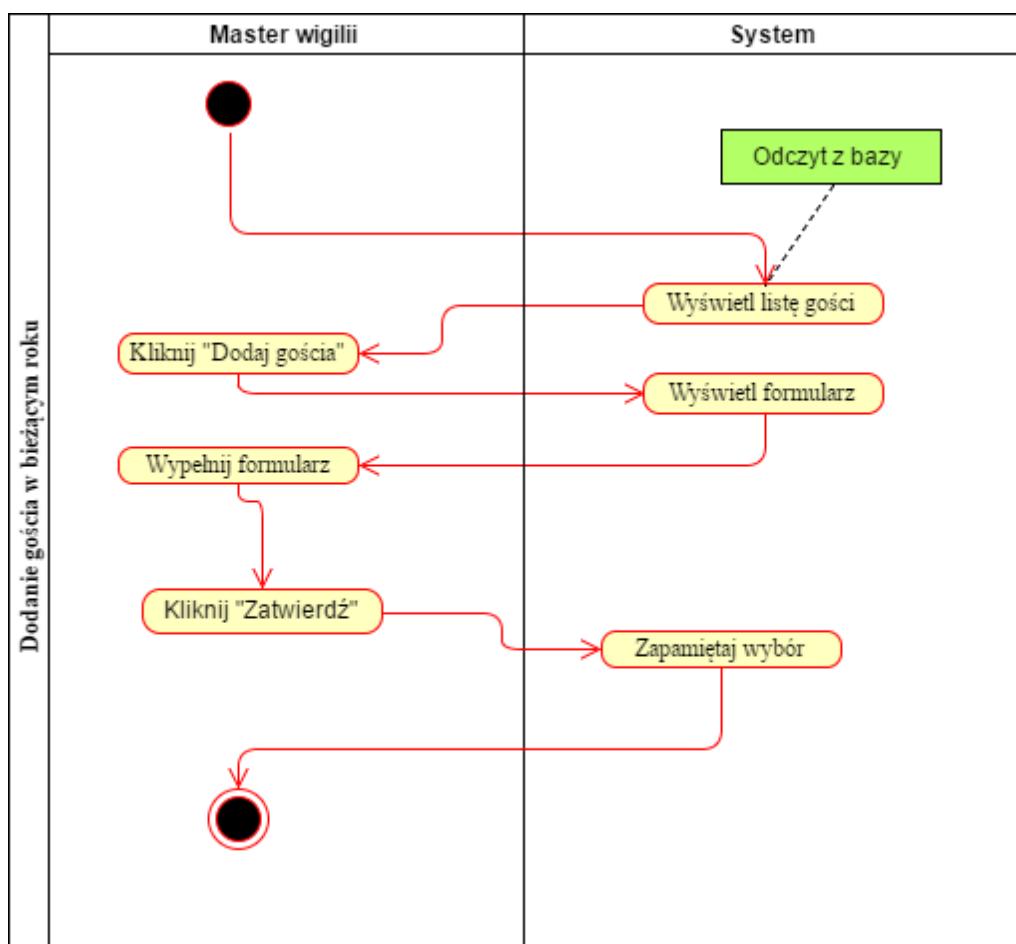
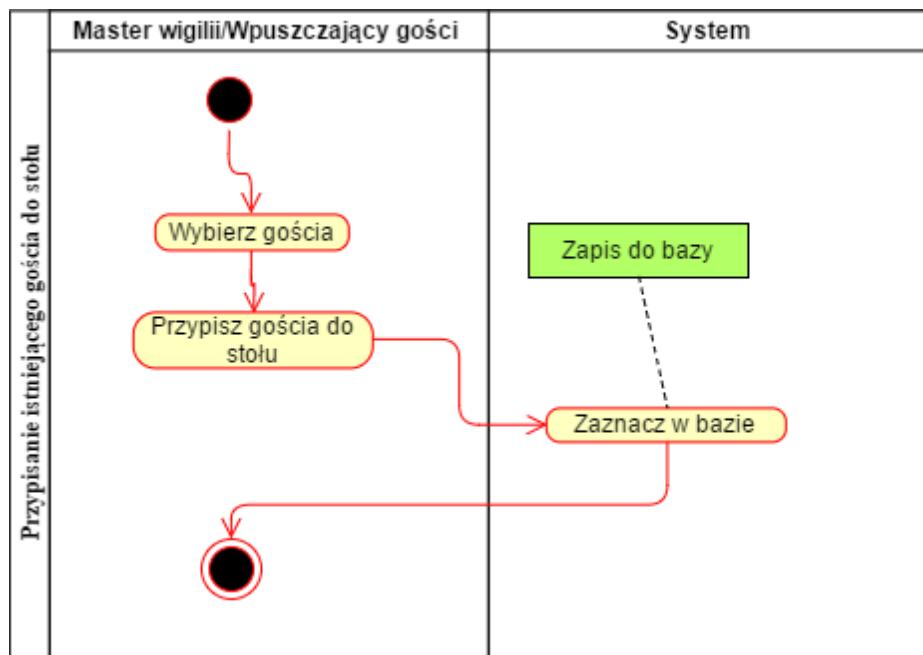
## 5. Diagram czynności

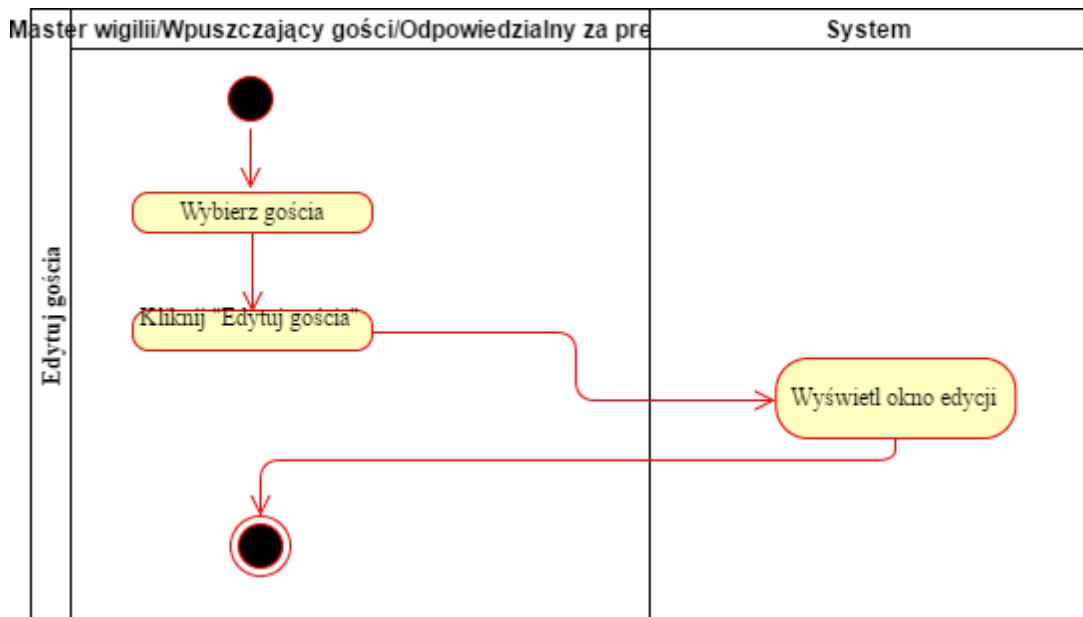
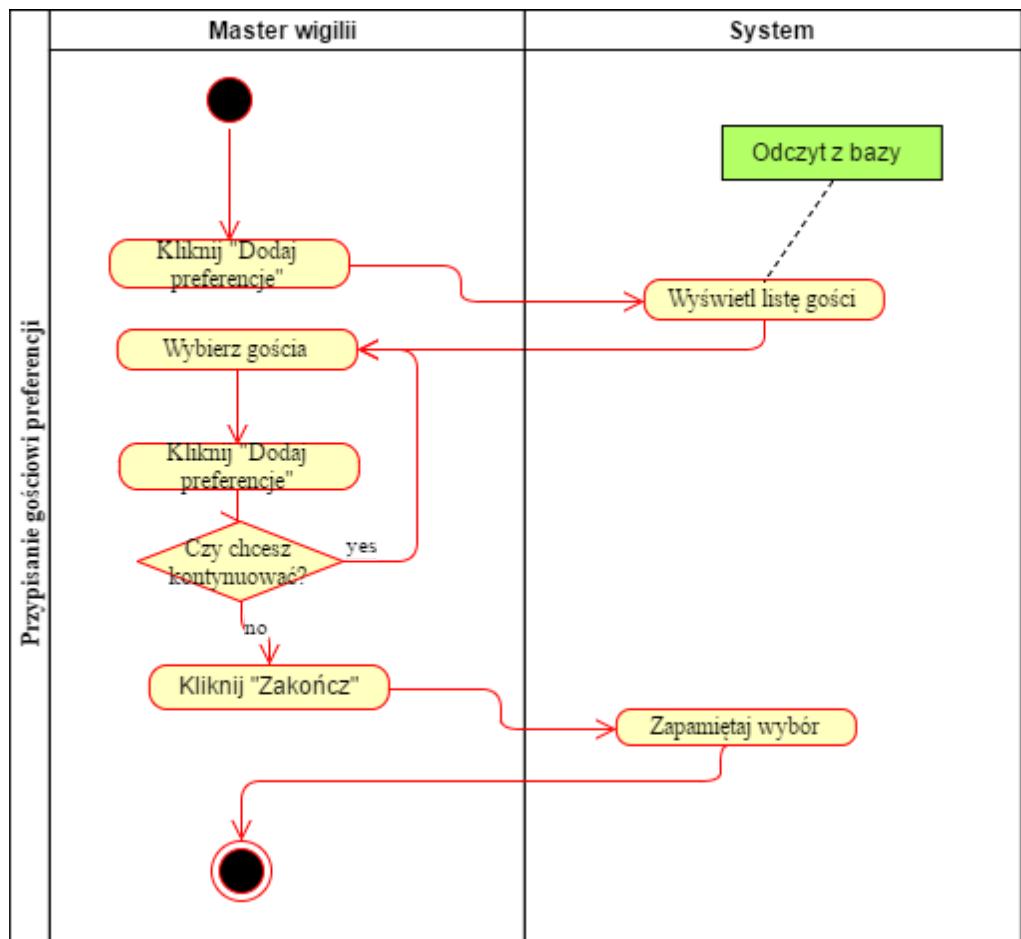


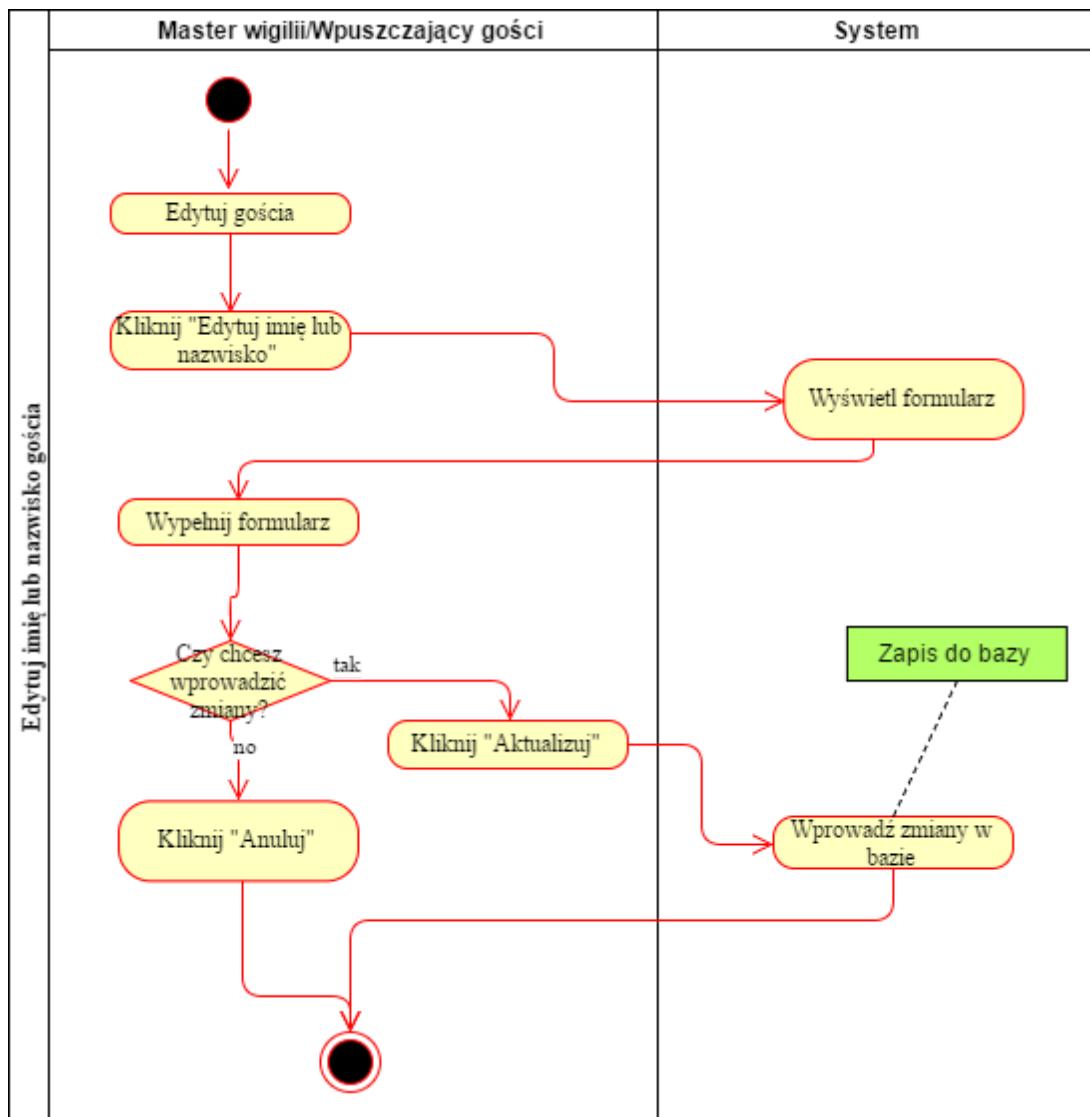


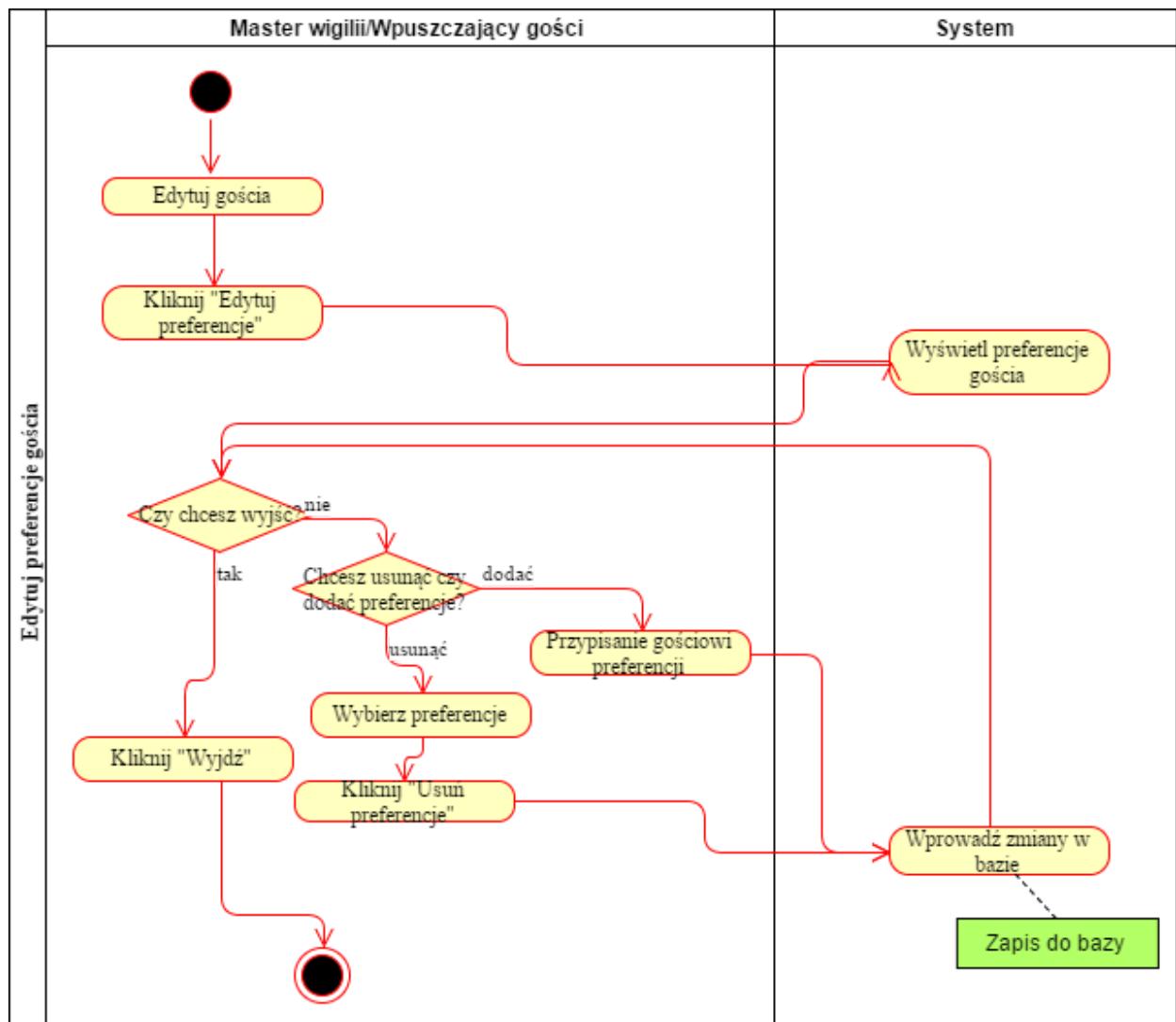


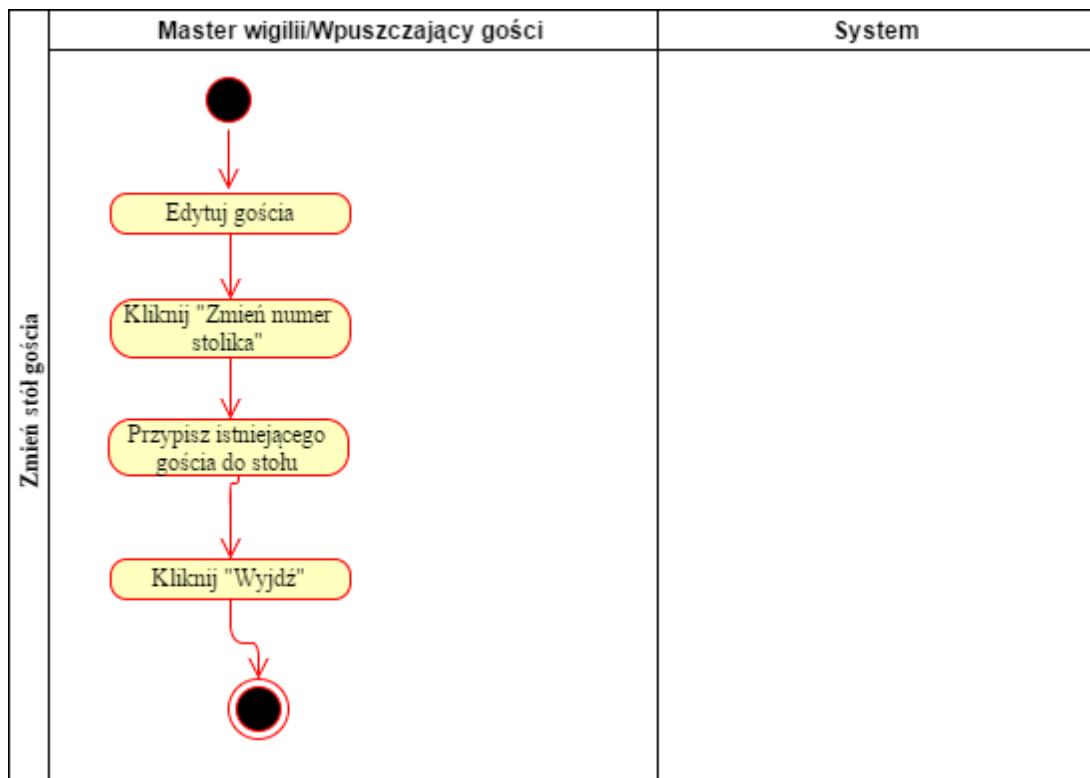


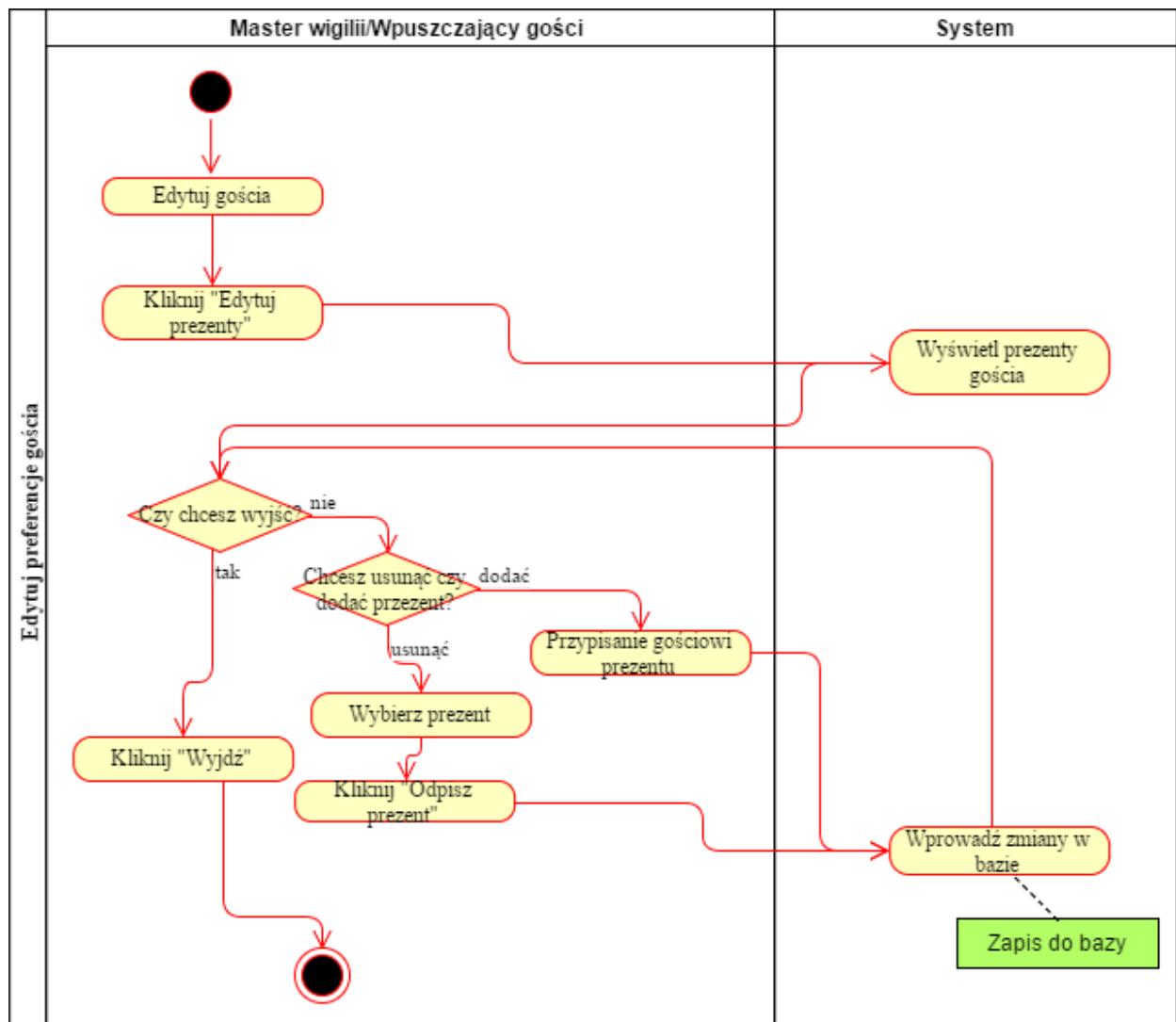




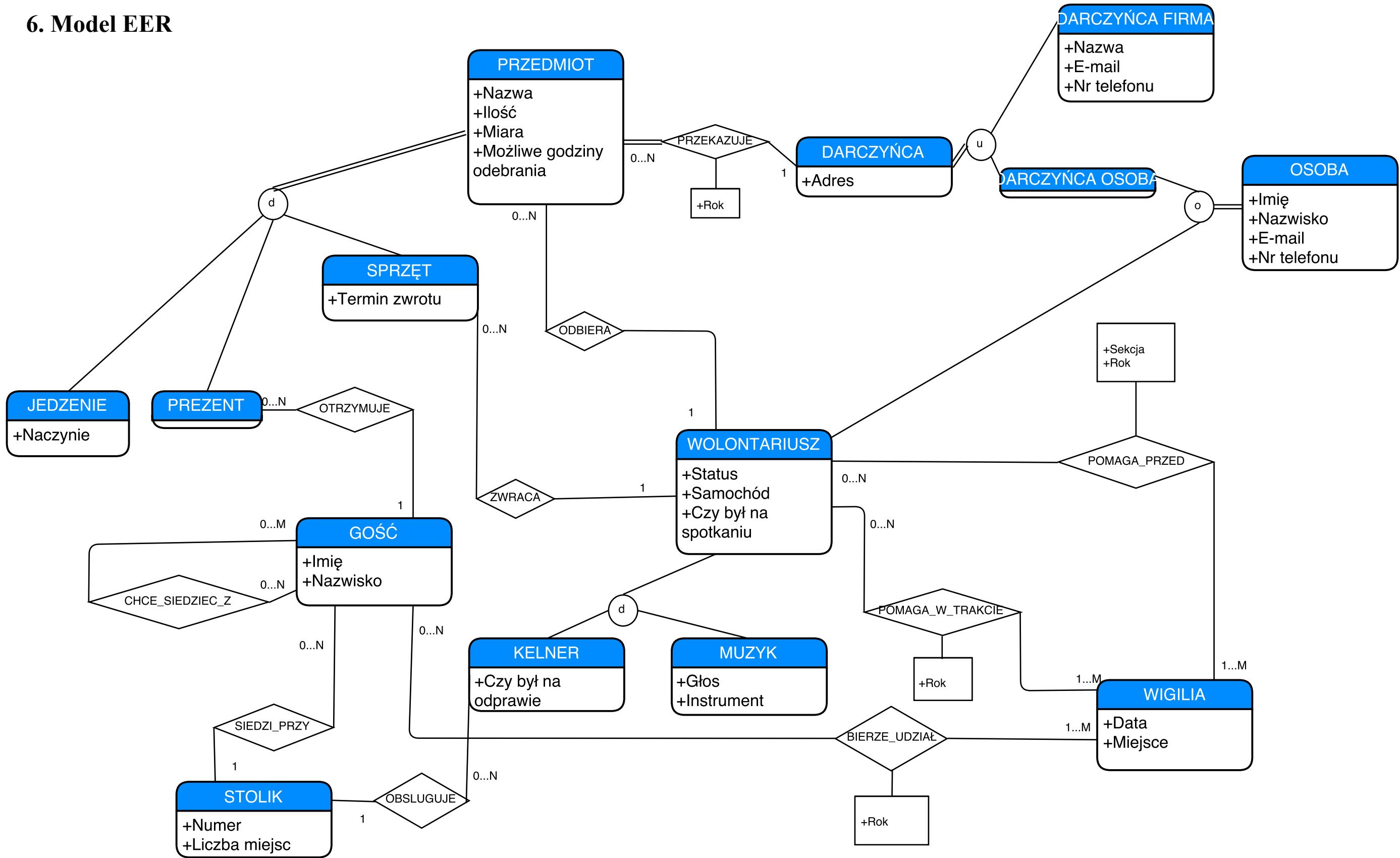




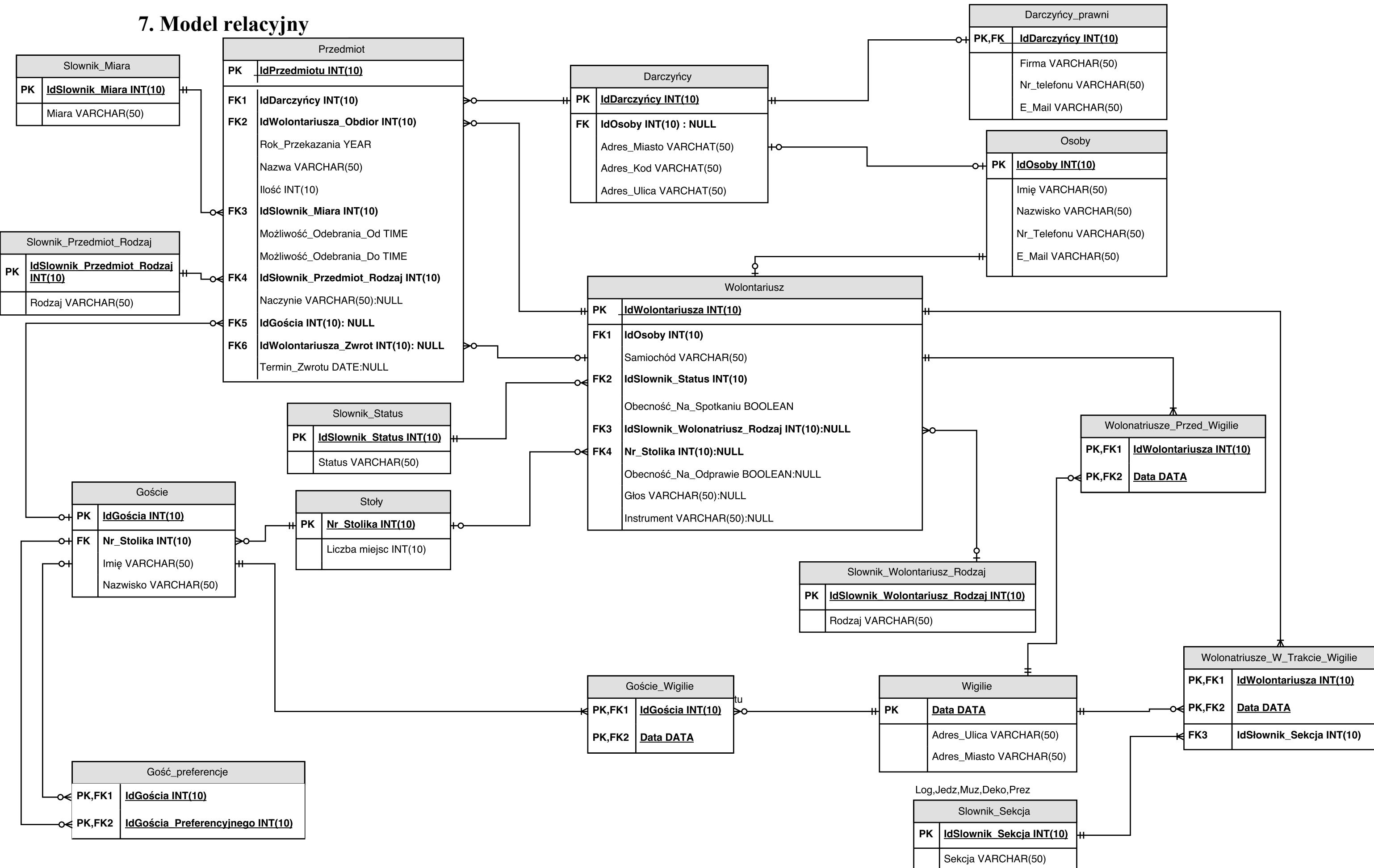




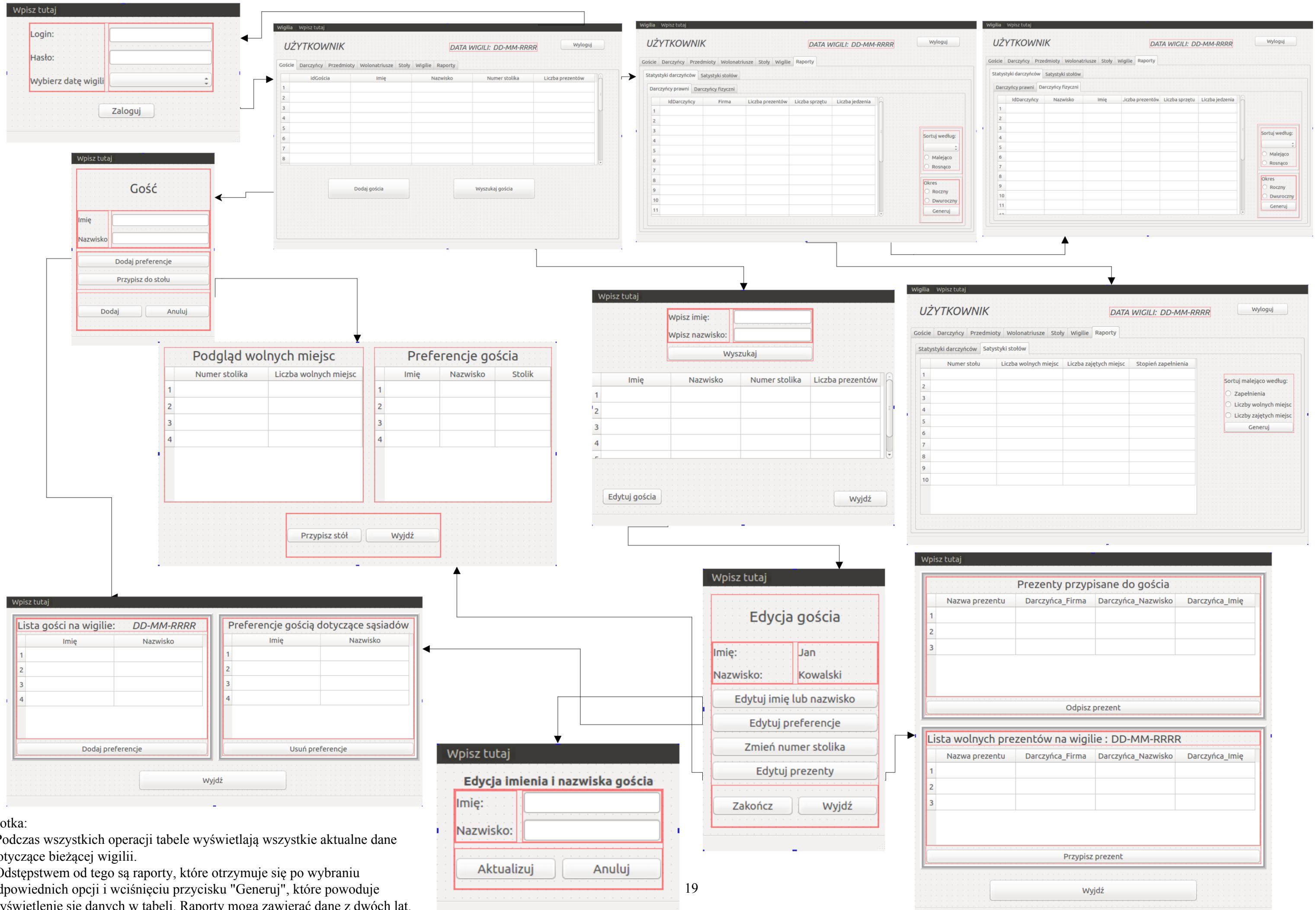
## 6. Model EER



## 7. Model relacyjny



# 8. Projekt GUI



## 9. Projekty raportów

### Statystyki darczyńców

Raport prezentuje liczbę przekazanych przedmiotów przez darczyńców, w danym okresie czasu-roczne, dwuroczne. Możliwe jest wybranie rodzaju przedmiotu: jedzenie, prezenty, sprzęt. Możliwe jest wybranie opcji sortowania. Raport został podzielony na darczyńców indywidualnych i prawnych

### Statystyki stołów

Raport przedstawia listę stołów z liczbą osób przypisanych do stołu, a także z liczbą wolnych miejsc. Dla każdego stołu liczny jest stopień zapełnienia stołu. Możliwe jest sortowanie po każdym z przedstawionych parametrów.

## 10. Implementacja bazy danych

Implementację bazy danych zrealizowano w systemie zarządzania relacyjnymi bazami danych MySQL. Korzystając z programu MySQL Workbench 6.3 CE, na podstawie zaprojektowanego modelu relacyjnego, wygenerowano skrypty służące do utworzenia bazy na serwerze MySQL. Napisano także dodatkowe skrypty uzupełniające bazę przykładowymi danymi, aby zapewnić kontrolowane warunki na czas testów aplikacji i możliwość przywrócenia bazy do pierwotnego stanu po wystąpieniu błędów.

## 11. Implementacja GUI

Interfejs graficzny użytkownika wykonano w języku C++ korzystając z biblioteki Qt 5.7 oraz środowiska programistycznego Qt Creator 4.2.0. Bibliotekę Qt wybrano ze względu na dobre wsparcie dla interakcji z serwerem MySQL. Udostępnia ona funkcje służące do logowania do bazy, wysyłania zapytań i łatwej modyfikacji zapytań w zależności od stanu zmiennych w programie. Wyniki zapytania można w łatwy sposób wyświetlić w interfejsie graficznym w postaci tabeli QTableView.

Ważnym elementem implementacji aplikacji, jest klasa Gosc o następującej strukturze

```
class Gosc
{
public:
    // tworzy pustego gościa
    Gosc();
    // tworzy gościa pobierając dane z bazy wg id
    Gosc(int idGoscia);
    // tworzy kopię gościa ze wskaźnika
    Gosc(Gosc *to_copy);
```

```

    // zwraca fałsz gdy dane do przesłania są niekompletne
    bool send_to_database();
    // wyświetla zawartość obiektu
    void print();

    bool new_guest;
    int id;
    QString first_name;
    QString last_name;
    int table;
    QVector<int> preferences;
    QVector<int> gifts;
};


```

Pola klasy pozwalają na przechowywanie wszystkich informacji o gościu potrzebnych w procesie jego dodawania lub edycji. Zamknięcie tych informacji w jednej klasie umożliwia łatwe przekazywanie całego obiektu między oknami. Dodatkowo pracę ułatwia konstruktor Gosc(int idGoscia), który przyjmując id jako argument tworzy obiekt, którego pola są uzupełnione informacjami na temat danego gościa pobranymi z bazy. Kolejnym usprawnieniem jest metoda send\_to\_database(), która umożliwia szybkie przekazanie danych z obiektu wprost do bazy danych. Wykonuje ona odpowiednio dodanie nowych krotek lub modyfikację istniejących w zależności od stanu bazy.

## Zakładka “Goście”

W tej zakładce znajduje się lista gości oraz dwa przyciski: “Dodaj gościa” i “Wyszukaj gościa”. Zgodnie z opisami, pierwszy przycisk przenosi użytkownika do okna dodawania gościa, a drugi przycisk kieruje go do okna wyszukiwania gościa. Zawartość listy gości jest odświeżana przed każdym wyświetlaniem okna głównego. Znajdują się w niej następujące informacje o każdym z gości: id, imię, nazwisko, numer stolika oraz liczba prezentów. Wyświetlanie danych jest realizowane poprzez wysłanie do bazy zapytania

```

SELECT
    idGoscia AS 'idGościa',
    imie AS 'Imię',
    nazwisko AS 'Nazwisko',
    Nr_Stolika AS 'Numer stolika',
    l_prezentow AS 'Liczba prezentów'
FROM
    goscie
    LEFT JOIN
    (SELECT
        idGoscia, COUNT(*) AS l_prezentow
    FROM
        przedmiot
    WHERE
        idSlownik_Przedmiot_Rodzaj = 1
    GROUP BY idGoscia) AS l USING (idGoscia)
ORDER BY (idGoscia);

```

## Okno dodawania gościa

Wyświetlenie tego okna powoduje utworzenie nowego obiektu typu Gosc, w którym będą zapisywane informacje wprowadzane przez użytkownika. W oknie znajdują się dwa pola tekstowe umożliwiające wpisanie imienia i nazwiska gościa oraz cztery przyciski: “Dodaj preferencje”, “Przypisz do stołu”, “Dodaj” oraz “Anuluj”. Dwa pierwsze odsyłają użytkownika do odpowiednich okien. Kliknięcie przycisku “Dodaj” powoduje sprawdzenie, czy wszystkie wymagane dane o gościu zostały wprowadzone, jeśli tak, następuje wywołanie metody add\_to\_database() na obecnie przetwarzanym obiekcie Gosc i powrót do okna głównego. Przycisk “Anuluj” powoduje usunięcie utworzonego obiektu i powrót do okna głównego.

## Okno przypisania do stołu

W oknie wyświetlane są dwie tabele: “Podgląd wolnych miejsc” i “Preferencje Gościa”. Pierwsza wyświetla informacje zwrotne z zapytania

```
SELECT
    nr_stolika AS 'Numer stolika',
    liczba_miejsc - COUNT(*) AS 'Liczba wolnych miejsc'
FROM
    goscie
    JOIN
        stoly USING (nr_stolika)
GROUP BY nr_stolika;
```

Druga wyświetla dane o gościach z którymi chciałby siedzieć gość, którego informacje są obecnie przetwarzane. Informacje są pobierane z bazy poprzez serię zapytań wygenerowanych na podstawie zawartości pola gosc.preferences typu QVector<int>, zawierającego wszystkie id gości preferowanych.

## Okno edycji preferencji

Okno zawiera dwie tabele: “Lista gości” oraz “Preferencje”. Dodanie gościa z listy do preferencji polega na zaznaczeniu, przez kliknięcie, odpowiedniego rzędu z listy i kliknięcie przycisku “Dodaj preferencję”. Usunięcie preferencji wykonuje się analogicznie, przez zaznaczenie pozycji w prawej tabeli i kliknięcie przycisku “Usuń preferencję”. Zawartość tabeli z listą gości jest odświeżana w momencie wyświetlania okna, przez zapytanie:

```
SELECT
    imie AS 'Imię', nazwisko AS 'Nazwisko', idGoscia
FROM
    goscie;
```

Kolumna idGoscia jest ukrywana, jej pobranie z bazy jest wykonane dla potrzeb identyfikacji gości wybranych do dodania do preferencji.

Druga tabela jest zapełniana za pomocą serii prostych zapytań pobierających dane o gościach, których id są przechowywane w polu gosc.preferences.

## Okno “Wyszukaj gościa”

W tym oknie użytkownik dostaje możliwość wyszukania gości po ich imieniu lub nazwisku lub używając obu pól. Przycisk “wyszukaj” lub klawisz “Enter” generuje listę gości pasujących do wpisanych danych. Zaznaczając odpowiedniego gościa i klikając “Edytuj gościa” program otwiera okno Edycji gościa. Jednocześnie tworzy tymczasowy obiekt, który odzwierciedla dane znajdujące się w bazie dotyczące tej osoby.

```
SELECT
    IdGoscia AS IdGościa,
    imie AS 'Imię', Nazwisko,
    Nr_Stolika AS 'Numer stolika',
    I_prezentow AS 'Liczba prezentów'
FROM
    goscie
LEFT JOIN
    (SELECT idGoscia, count(*) AS I_prezentow FROM przedmiot
     WHERE
        (idSlownik_Prezdmiot_Rodzaj = 1) GROUP BY idGoscia)
    AS I USING (idGoscia)
    WHERE (imie = "andrzej" AND nazwisko = "ziółek");
```

## Okno “Edycja gościa”

Jest to ogólne okno edycji gościa. Z tego poziomu, użytkownik ma dostęp do okien odpowiedzialnych za: edycje imienia i nazwiska, preferencje co do osób, zmiany przypisanego stolika i prezentów.

Przycisk “OK” wprowadza zmiany w bazie, bazując na utworzonym i zmodyfikowanym obiekcie.

Przycisk “Anuluj” nie wprowadza żadnych zmian do bazy i zamyka okno edycji gościa.

Część kodu implementującego zmiany w bazie danych(bazując na tymczasowym obiekcie), po wciśnięciu przycisku “OK”.

```
query.exec("SET FOREIGN_KEY_CHECKS=0;");
query.prepare("REPLACE INTO goscie VALUES (?, ?, ?, ?, ?);");
query.addBindValue(id);
query.addBindValue(table);
query.addBindValue(first_name);
query.addBindValue(last_name);
success &= query.exec();
query.exec("SET FOREIGN_KEY_CHECKS=1");
```

## Okno “Edycja imienia lub nazwiska”

W oknie tym użytkownik ma możliwość wprowadzenia zmian w polach Imię i Nazwisko obiektu gość.

Przycisk “OK” przenosi do okna Edycji gościa, wprowadzając zmiany do obiektu.

Przycisk “Anuluj” nie wprowadza żadnych zmian do bazy i zamyka okno edycji gościa.

## **Okno “Edycja preferencji”**

Tutaj użytkownik zmienia preferencję gościa co do osób z którymi dany gość wyraża chęć spędzenia wigilii.

Z listy gości można wybierać pojedyńcze osoby a następnie dodawać je do listy preferowanych osób klikając “Dodaj preferencje”. Z listy preferowanych gości usuwanie pozycji przebiega analogicznie. Listy nie zawierają tych samych elementów. Potwierdzenie przyciskiem “OK”.

## **Okno “Zmień numer stolika”**

Okno udostępnia możliwość zmiany przypisanego stolika do gościa.

W liście lewej znajdują się numery stolików wraz z ich wolnymi miejscami.

W liście prawej preferencję osób z którymi dany gość wyraża chęć spędzenia wigilii.

Przycisk “Przypisz stół” zatwierdza zmiany w obiekcie gościa.

Przycisk “Wyjdź” przenosi do okna Edycji gościa bez wprowadzania zmian.

## **Okno “Edytuj prezenty”**

W tym oknie użytkownik dostaje możliwość przypisania i odpisania prezentu do danego gościa. Lista górna przedstawia prezenty które już są przypisane. Lista dolna przedstawia prezenty które nie są przypisane do żadnego z gości. Poprzez zaznaczenie i kliknięcie odpowiednio w Odpisz prezent-kasujemy prezent przypisany do gościa, Przypisz prezent-przypisujemy zaznaczony wolny prezent do danego gościa.

Przycisk “OK” przenosi do okna Edycji gościa, wprowadzając zmiany do obiektu.

Przykładowe zapytanie pokazujące listę wolnych prezentów.

```
SELECT Nazwa AS 'Nazwa Prezentu',
IdPrzedmiotu AS IdPrzedmiotu,
firma as 'Firma',
0 AS 'IdDarczyńcy',
0 AS 'Nazwisko',
0 AS 'Imię'
FROM darczyncy, osoby, darczyncy_prawni
LEFT JOIN przedmiot ON idDarczyncy_Prawni = idDarczyncy_Przedmiot
WHERE (idGoscia IS NULL OR idGoscia=2) AND osoby.idOsoby=darczyncy.idOsoby
UNION
SELECT Nazwa AS 'Nazwa Prezentu',
IdPrzedmiotu AS IdPrzedmiotu,
0 as 'Firma',
idDarczyncy AS 'IdDarczyńcy',
Nazwisko,
Imie AS 'Imię'
FROM darczyncy JOIN osoby USING (idOsoby) LEFT JOIN przedmiot ON idDarczyncy = idDarczyncy_Przedmiot
WHERE (idGoscia IS NULL OR idGoscia=2);
```

## 12. Implementacja raportów

Dostęp do raportów można uzyskać w zakładce “Raporty” w oknie głównym aplikacji. Raporty są podzielone na dwie zakładki: “Statystyki darczyńców” i “Statystyki stołów”. W zakładce “Statystyki darczyńców” dodatkowo znajdują się dwie zakładki raportów dotyczących darczyńców fizycznych oraz darczyńców prawnych.

### Raport “Darczyńcy prawni”

Zawartość zakładki raportu o darczyńcach zawiera tabelę oraz panel po prawej stronie, w którym możliwa jest zmiana ustawień sortowania oraz okresu generowanego raportu. Kliknięcie na przycisk “Generuj” powoduje wysłanie do bazy zapytania o treści:

```
SELECT
    idDarczyncy_Prawni AS 'IdDarczyńcy',
    firma AS 'Firma',
    SUM(IF(idSlownik_Prezemiot_Rodzaj = 1
        AND (Rok_Przekazania = 2016
        OR Rok_Przekazania = 2017),
        1,
        0)) AS 'Liczba prezentów',
    SUM(IF(idSlownik_Prezemiot_Rodzaj = 2
        AND (Rok_Przekazania = 2016
        OR Rok_Przekazania = 2017),
        1,
        0)) AS 'Liczba sprzętu',
    SUM(IF(idSlownik_Prezemiot_Rodzaj = 3
        AND (Rok_Przekazania = 2016
        OR Rok_Przekazania = 2017),
        1,
        0)) AS 'Liczba jedzenia'
FROM
    darczyncy_prawni
    LEFT JOIN
    przedmiot ON idDarczyncy_Prawni = idDarczyncy_Prezemiot
GROUP BY idDarczyncy_Prawni
ORDER BY `Liczba sprzętu` DESC , `IdDarczyńcy`;
```

Przedstawione zapytanie jest zapytaniem przykładowym, wygenerowanym dla ustawień: sortowanie wg “Liczby sprzętu”, malejąco, dla okresu dwurocznego.

## Raport “Darczyńcy fizyczni”

Raport “Darczyńcy fizyczni” ma identyczne opcje i jest generowany analogicznie do raportu “Darczyńcy prawni”. Dotyczy on jednak innego zbioru darczyńców i zamiast nazw firm wyświetlane są imiona i nazwiska darczyńców. Poniżej przedstawiono przykładowe zapytanie do bazy, wygenerowane dla ustawień: sortowanie wg “Liczby prezentów”, malejąco, dla okresu dwurocznego.

```
SELECT
    idDarczyncy AS 'IdDarczyncy',
    Nazwisko,
    Imie AS 'Imię',
    SUM(IF(idSlownik_Prezemiot_Rodzaj = 1
        AND (Rok_Przekazania = 2016
        OR Rok_Przekazania = 2017),
        1,
        0)) AS 'Liczba prezentów',
    SUM(IF(idSlownik_Prezemiot_Rodzaj = 2
        AND (Rok_Przekazania = 2016
        OR Rok_Przekazania = 2017),
        1,
        0)) AS 'Liczba sprzętu',
    SUM(IF(idSlownik_Prezemiot_Rodzaj = 3
        AND (Rok_Przekazania = 2016
        OR Rok_Przekazania = 2017),
        1,
        0)) AS 'Liczba jedzenia'
FROM
    darczyncy
    JOIN
    osoby USING (idOsoby)
    LEFT JOIN
    przedmiot ON idDarczyncy = idDarczyncy_Prezemiot
GROUP BY idDarczyncy
ORDER BY `Liczba prezentów` DESC , `IdDarczyncy`;
```

## Raport “Statystyki stołów”

Raport przedstawia statystyki stołów znajdujących się na wigilii. Przedstawia dane o: Numerze stołu, liczby miejsc wolnych/zajętych i stopniu zapełnienia. Możliwa jest zmiana ustawień sortowania. Kliknięcie na przycisk “Generuj” powoduje wysłanie do bazy zapytania o treści podobnej do:

```
SELECT
    nr_stolika AS 'Numer stołu',
    liczba_miejsc - count(*) AS 'Liczba wolnych miejsc',
    count(*) AS 'Liczba zajętych miejsc',
    count(*)/Liczba_Miejsc AS 'Stopień zapełnienia'
FROM
    goscie
    JOIN
    stoly USING (nr_stolika)
GROUP BY nr_stolika
ORDER BY 'Stopień zapełnienia' DESC;
```