*Politechnika Wrocławska*
*Wydział Elektroniki*

BAZY DANYCH 2 – PROJEKT

---

# ZAKŁAD TRANSPORTOWY

---

*Autorzy:*

Alicja Myśliwiec 248867
Daria Milczarska 248894

Termin zajęć:
*Środa 11.15 – 13.00*

*Prowadzący:*

*Dr inż. Tomasz Babczyński*

# 1 Model konceptualny bazy danych

## 1.1 Opis świata rzeczywistego

- Firma transportowa zajmuje się przewozem rzeczy pojazdami samochodowymi i oferuje szeroką gamę rozwiązań transportowych,
- Przewozy towarów realizowane są środkami transportu drogowego przystosowanymi do przewozu chemii i kosmetyków, AGD i elektroniki, okien i mebli, dużych gabarytów typowych dla branży energetycznej, zabawek, towarów zamrożonych lub schłodzonych, towarów ciężkich, towarów masowych, włączając transport cysternami mleka z gospodarstw rolnych, samochodów osobowych, pojazdów ciężarowych i autobusów z wykorzystaniem specjalnych naczep, odpadów i materiałów wtórnych,
- W przypadku niektórych towarów będą konieczne dodatkowe opłaty za użycie specjalnego sprzętu. Do takich towarów należą m.in. materiały niebezpieczne,
- Dodatkową opłatą obciążone będą również transporty poza granice kraju,
- W firmie pracuje wykwalifikowana kadra składająca się z 15 osób: 10 kierowców, 2 mechaników, kierownika, administratora oraz właściciela, który jest jednocześnie kierownikiem,
- Z racji wysokiej jakości usług, firma ma duże perspektywy rozwoju, w niedalekiej przyszłości planowane jest wprowadzenie transportu osób i zwierząt,
- Aplikacja bazodanowa firmy będzie umożliwiała prowadzenie dokumentacji w formie elektronicznej, co usprawni zarządzanie zasobami oraz integrację z innymi systemami np. bazą danych firm zajmujących się sprzedażą samochodów, pojazdów ciężarowych i cystern,
- Do dokumentacji należą m.in. dane pojedynczych transportów, faktury, raporty, dane pracowników oraz dane klientów.

### 1.1.1 Klient

Osoba, która decyduje się na zamówienie usługi transportowej, powinna mieć możliwość:

- Sprawdzenia oferty firmy i cennika usług
- Sprawdzenia dostępności pojazdów
- Utworzenia konta
- Usunięcia konta
- Zalogowania się do panelu klienta
- Złożenia zamówienia
- Anulowania zamówienia
- Sprawdzenia szczegółów swojego zamówienia
- Wyświetlenia historii swoich zamówień

Aby wszystkie funkcje były dostępne, klient musi mieć założone konto. Jeżeli klient chce zarezerwować transport na określony termin, to pojazd musi być wolny i dostępny kierowca do danego pojazdu.

### 1.1.2 Kierownik

Zadaniem kierownika jest sprawdzanie, czy klient złożył zamówienie poprzez panel klienta, czy w inny sposób np. telefonicznie. W drugim przypadku kierownik musi ręcznie utworzyć zamówienie oraz przypisać do niego dane klienta. Kierownik pobiera opłaty, wystawia faktury oraz generuje raporty i statystyki. Zmiana etapów transportu również należy do obowiązków kierownika.

### 1.1.3 Kierowca

Kierowca odpowiedzialny jest za dostarczenie towarów w określone miejsce w określonym czasie. Musi on posiadać uprawnienia do kierowania pojazdem, w niektórych wypadkach specjalne uprawnienia np. do przewozu towarów niebezpiecznych oraz aktualne badania. Generalnie za dany transport odpowiedzialny jest jeden kierowca, chyba że transport jest wielogodzinny – w takim wypadku przydzieleni są dwaj kierowcy.

### 1.1.4 Mechanik

Mechanik odpowiedzialny jest za kontrolowanie stanu technicznego pojazdów, uzupełnianie płynów chłodzących, hamulcowych, do spryskiwaczy i oleju, dokonywanie napraw usterek oraz wykonywanie regularnych przeglądów pojazdów. Kontroluje on stan magazynu z częściami oraz płynami do pojazdów. Wyposaża pojazdy o niezbędne przedmioty takie jak apteczka czy płyn do dezynfekcji.

### 1.1.5 Właściciel

Właściciel posiada wyższy poziom uprawnień. Jest on również kierownikiem, dlatego posiada uprawnienia i funkcje należące do grupy uprawnień oraz funkcji kierownika. Właściciel wprowadza do systemu dane o pracownikach, nadaje im odpowiednie uprawnienia, zamawia potrzebne towary oraz ustala ceny usług.

### 1.1.6 Administrator

Jest to osoba z uprawnieniami właściciela, jednak jej głównym zadaniem jest zarządzanie bazą danych od strony technicznej. Do jego obowiązków należy robienie kopii zapasowych bazy, dodawanie kont właścicieli, zmiana parametrów bazy danych i naprawianie błędów systemowych.

### 1.1.7 Współpraca z systemami zewnętrznymi

Po stworzeniu jednolitego systemu wymiany danych, system bazodanowy może ułatwić kontakt z systemami zewnętrznymi. Wzajemnie wymienianie się informacjami np. z hurtownią płynów samochodowych mogłoby zniwelować możliwość wyczerpania się któregoś z płynu w magazynie poprzez automatyczne składanie zamówień.

## 1.2 Wymagania funkcjonalne i niefunkcjonalne systemu

### 1.2.1 Wymagania funkcjonalne

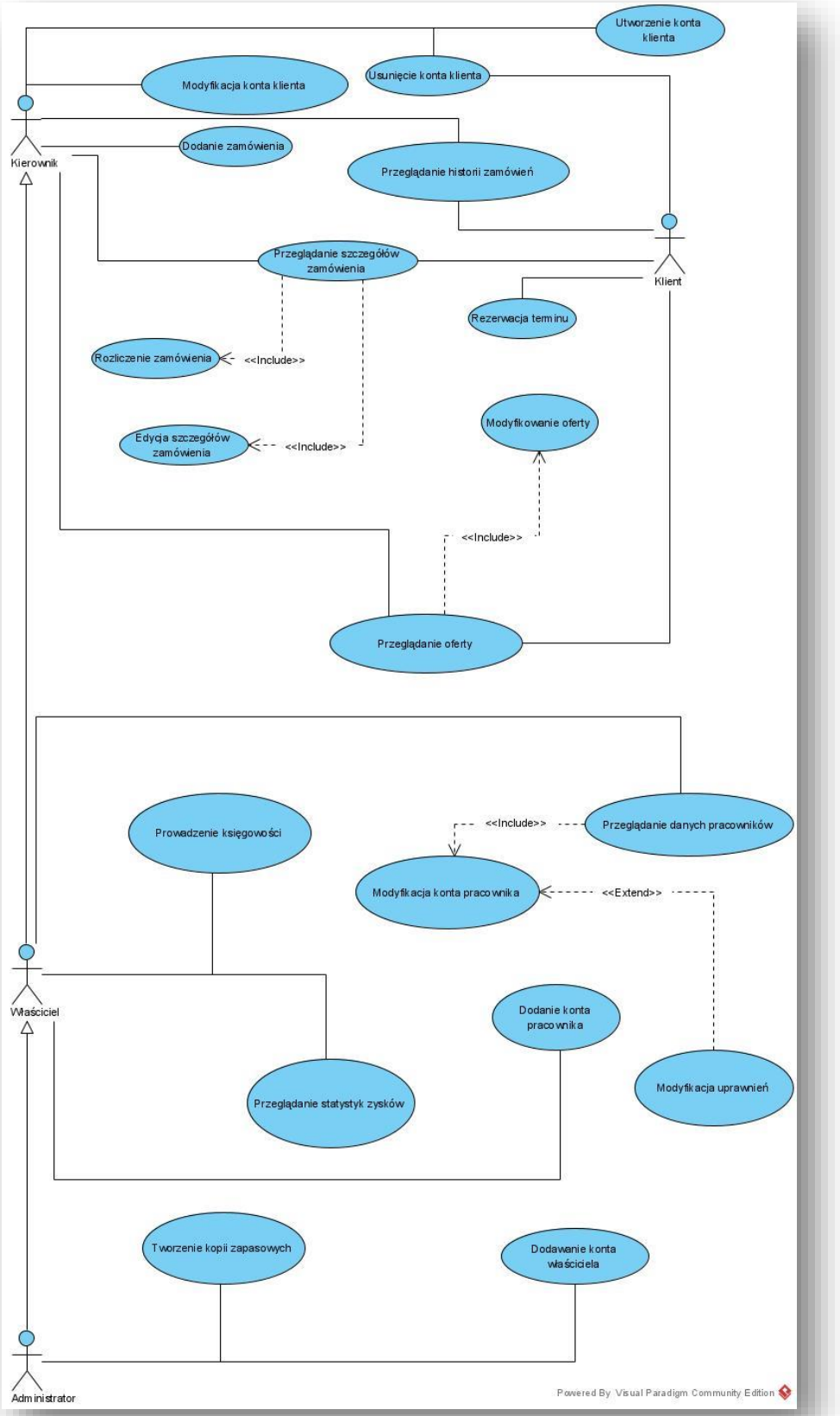
- Rejestracja użytkownika systemu (utworzenie konta)
- Usunięcie konta
- Utworzenie zamówienia
- Anulowanie zamówienia
- Sprawdzenie oferty firmy i cennika usług
- Sprawdzenia dostępności pojazdów
- Sprawdzenie szczegółów zamówienia
- Wyświetlenie historii zamówień
- Zmiana szczegółów zamówienia
- Generowanie raportów i statystyk
- Przechowywanie danych o pracownikach
- Przeglądanie, dodawanie, edytowanie oraz usuwanie danych przez użytkownika jedynie w ramach swoich uprawnień

### 1.2.2 Wymagania niefunkcjonalne

- **Odporność na utratę danych** - archiwizacja stanu bazy danych, możliwość wczytania bazy danych z pliku, tworzenie kopii zapasowych,
- **Bezpieczeństwo i poufność danych** - dostęp do danych jest ograniczony, każdy użytkownik ma inne, indywidualne uprawnienia poprzez osobne konta,
- **Wydajność** - system musi działać szybko, a maksymalny czas na wykonanie zapytania nie powinien przekraczać 7 sekund. Wyjątek stanowią operacje wykonywanie sporadycznie (np. generowanie rocznych raportów). Założenia projektowe powinny być spełnione dla liczby pracowników nie większej do 20 i liczby klientów nie większej niż 2000,
- **Skalowalność** - możliwość rozbudowy aplikacji i bazy danych o inne funkcjonalności, możliwość dołączenia modułów, które w przyszłości pozwolą na współpracę z innymi systemami zewnętrznymi lub dostosowanie się do nowych wymagań regulowanych zmieniającymi się przepisami lub sytuacją,
- **Komunikacja z innymi systemami bazodanowymi** - system powinien umożliwiać automatyczną wymianę danych z innymi systemami np. bazy danych hurtowni z produktami samochodowymi, mile widziane będą powiązania z bazami danych innych partnerów biznesowych.

## 1.3 Specyfikacja wymagań funkcjonalnych za pomocą diagramu przypadków użycia

### 1.3.1 Diagram przypadków użycia




**Rysunek 1: Diagram przypadków użycia**

### 1.3.2 Opis słowny przypadków użycia

**1. Utworzenie konta klienta:**

1.1. Cel:

Utworzenie w bazie danych firmy rekordu odpowiadającego za przetrzymywanie historii transakcji klienta oraz przetrzymywanie jego danych osobowych.

1.2. Warunek początkowy:

Warunkiem koniecznym jest pojawienie się w firmie nowego klienta.

1.3. Warunek końcowy:

Kompletne wypełnienie formularza rejestracyjnego.

1.4. Realizacja:

- **KIEROWNIK** wybiera opcję DODAJ KLIENTA
- Wypełnienie formularza i weryfikacja prawdziwości danych
- Umieszczenie w bazie danych rekordu z danymi klienta

**2. Usunięcie konta klienta:**

2.1. Cel:

Usunięcie z bazy danych wszystkich danych związanych z danym klientem oprócz tych potrzebnych do księgowości.

2.2. Warunek początkowy:

Posiadanie konta.

2.3. Warunek końcowy:

W bazie nie ma już rekordu zawierającego dane klienta.

2.4. Realizacja:

- **KLIENT** lub **KIEROWNIK** (na prośbę klienta) wybiera opcję USUŃ KONTO
- Potwierdzenie operacji przyciskiem ZATWIERDŹ
- Usunięcie z bazy danych rekordu zawierającego informację o kliencie

**3. Modyfikacja konta klienta:**

3.1. Cel:

Zmiana danych klienta. Może być związana ze zmianą numeru telefonu, nazwiska czy adresu zamieszkania.

3.2. Warunek początkowy:

Istnienie konta.

3.3. Warunek końcowy:

Dane po aktualizacji odpowiadają stanowi rzeczywistemu.

3.4. Realizacja:

- **KIEROWNIK** wybiera opcję MODYFIKACJA DANYCH KLIENTA
- Wypełnienie formularza
- Zatwierdzenie zmian przyciskiem
- Aktualizacja już istniejącego rekordu w bazie danych

**4. Przeglądanie historii zamówień:**

4.1. Cel:

Klient lub kierownik może uzyskać informację na temat poprzednich zamówień klienta.

4.2. Warunek początkowy:

Klient wcześniej korzystał z usług firmy.

4.3. Warunek końcowy:

Uzyskanie przez klienta lub kierownika informacji, których potrzebuje.

4.4. Realizacja:

- **KLIENT** wybieraj opcję PRZEGLĄDAJ HISTORIĘ lub **KIEROWNIK** wybiera opcję HISTORIA KLIENTA
- Wyświetlenie informacji na temat poprzednich zamówień

**5. Dodanie zamówienia:**

5.1. Cel:

Utworzenie w bazie danych rekordu odpowiadającego nowemu zamówieniu.

5.2. Warunek początkowy:

Nowe zamówienie złożone osobiście, telefonicznie, mailowo itp.

5.3. Warunek końcowy:

Utworzenie nowego rekordu w bazie z nowym zamówieniem

5.4. Realizacja:

- **KIEROWNIK** wybiera opcję DODAJ ZAMÓWIENIE
- Wypełnienie formularza
- Utworzenie w bazie danych rekordu zawierającego informację o nowym zamówieniu

**6. Przeglądanie szczegółów zamówienia:**

6.1. Cel:

Sprawdzenie statusu zamówienia (rezerwacja terminu – potwierdzone – w trakcie – opcjonalnie: oczekiwanie na dostępne pojazdy - zakończone) .

6.2. Warunek początkowy:

Istnienie zamówienia.

6.3. Warunek końcowy:

Sprawdzenie statusu.

6.4. Realizacja:

- **KIEROWNIK** lub **KLIENT** wybiera opcję SZCZEGÓŁY ZAMÓWIENIA
- Wyszukanie konkretnego zamówienia na podstawie daty złożenia lub id klienta
- Wyświetlenie szczegółów danego zamówienia
- Zakończenie przeglądania przyciskiem POWRÓT

**KIEROWNIK** z tego poziomu ma możliwość edycji szczegółów zamówienia lub rozliczenia danego zamówienia.

**7. Edycja szczegółów zamówienia:**

7.1. Cel:

Umieszczenie w bazie danych informacji o aktualnym stanie zamówienia, tak, aby pracownicy oraz klienci mogli uzyskać tę informację.

7.2. Warunek początkowy:

Klient złożył zamówienie w firmie.

7.3. Warunek końcowy:

Zaakceptowanie wprowadzonej zmiany przez kierownika.

7.4. Realizacja:

- **KIEROWNIK** sprawdzając szczegóły zamówienia klienta wybiera opcję ZMIEŃ STATUS ZAMÓWIENIA
- Wybranie opcji z wysuwanej listy
- Zaakceptowanie zmian przyciskiem ZATWEIRDŹ
- Aktualizacja już istniejącego rekordu w bazie danych

**8. Rozliczenie zamówienia:**

8.1. Cel:

Wystawienie klientowi rachunku lub faktury za wykonanie usługi. Zmiana statusu zamówienia na zakończone.

8.2. Warunek początkowy:

Istnieje rozpoczęte, ale niezakończone zamówienie, które zostało już zrealizowane.

8.3. Warunek końcowy:

Zakończenie wszystkich czynności związanych z zamówieniem.

8.4. Realizacja:

- **KIEROWNIK** sprawdzając szczegóły zamówienia klienta wybiera opcję ROZLICZ ZAMÓWIENIE
- Wydrukowanie faktury lub paragonu
- Zmiana statusu zamówienia na zakończone
- Aktualizacja już istniejącego rekordu w bazie

**9. Rezerwacja terminu:**

9.1. Cel:

Klient wybiera sobie termin realizacji zamówienia.

9.2. Warunek początkowy:

Istnienie wolnych terminów.

9.3. Warunek końcowy:

Potwierdzenie zamówienia przez kierownika.

9.4. Realizacja:

- **KLIENT** wybiera opcję REZERWACJA TERMINU
- **KIEROWNIK** otrzymuje powiadomienie i potwierdza dostępność lub proponuje inny termin
- **KLIENT** akceptuje propozycję innego terminu lub ją odrzuca

**10. Przeglądanie oferty.**

10.1. Cel:

Uzyskanie informacji o aktualnej ofercie firmy.

10.2. Warunek początkowy:

Istnienie oferty.

10.3. Warunek końcowy:

Zakończenie przeglądania poprzez naciśnięcie przycisku.

10.4. Realizacja:

- **KLIENT** lub **KIEROWNIK** wybiera opcję ZOBACZ OFERTĘ
- Wyświetlenie aktualnej oferty
- Zakończenie przeglądania przyciskiem POWRÓT

**KIEROWNIK** z tego poziomu ma możliwość zmodyfikowania oferty.

## 11. Modyfikowanie oferty.

11.1. Cel:

Zmiana szczegółów oferty np. ceny.

11.2. Warunek początkowy:

Istnienie oferty. Aby móc zmienić szczegóły oferty, trzeba ją najpierw przejrzeć.

11.3. Warunek końcowy:

Dane w bazie po aktualizacji odpowiadają stanowi rzeczywistemu.

11.4. Realizacja:

- **KIEROWNIK** przeglądając ofertę wybiera opcję MODYFIKUJ OFERTĘ
- Wypełnienie formularza
- Aktualizacja danych

## 12. Prowadzenie księgowości:

12.1. Cel:

Zautomatyzowanie procesu prowadzenia księgowości w firmie.

12.2. Warunek początkowy:

Prowadzenie działalności.

12.3. Warunek końcowy:

Rozliczenie wszystkich zamówień.

12.4. Realizacja:

- **WŁAŚCICIEL** wybiera opcję ROZLICZENIA
- Wygenerowanie dokumentu prezentującego łączny podatek VAT do zapłacenia
- Zakończenie przeglądania przyciskiem POWRÓT

**13. Dodanie konta pracownika:**

13.1. Cel:

Dodanie danych pracownika do bazy.

13.2. Warunek początkowy:

Zatrudnienie nowego pracownika do firmy.

13.3. Warunek końcowy:

Kompletne wypełnienie formularza.

13.4. Realizacja:

- **WŁAŚCICIEL** wybiera opcję DODAJ PRACOWNIKA
- Wypełnienie formularza i weryfikacja prawdziwości danych
- Umieszczenie w bazie danych rekordu z danymi pracownika

**14. Modyfikacja konta pracownika:**

14.1. Cel:

Zmiana danych pracowników w bazie.

14.2. Warunek początkowy:

Istnienie kont pracowników w bazie.

14.3. Warunek końcowy:

Dane po aktualizacji odpowiadają stanowi rzeczywistemu.

14.4. Realizacja:

- **WŁAŚCICIEL** przeglądając dane o pracownikach wybiera opcję MODYFIKACJA DANYCH PRACOWNIKA
- Wypełnienie formularza
- Zatwierdzenie zmian przyciskiem ZATWIERDŹ
- Aktualizacja już istniejącego rekordu w bazie danych

Z tego poziomu **WŁAŚCICIEL** ma możliwość zmodyfikowania uprawnień pracownika.

## 15. Modyfikacja uprawnień:

15.1. Cel:

Przekazanie kierownikowi uprawnień właściciela.

15.2. Warunek początkowy:

Baza danych posiada przynajmniej po jednym koncie w właściciela i kierownika.

15.3. Warunek końcowy:

Wypełnienie formularza i zaakceptowanie operacji.

15.4. Realizacja:

- **WŁAŚCICIEL** przy modyfikacji danych o pracowniku może wybrać opcję UPRAWNIENIA
- Zmiana uprawnień
- Zatwierdzenie operacji przyciskiem ZATWIERDŹ
- Aktualizacja istniejącego rekordu w bazie

## 16. Przeglądanie danych pracowników:

16.1. Cel:

Uzyskanie informacji o pracownikach.

16.2. Warunek początkowy:

Baza danych zawiera konta pracowników i właścicieli.

16.3. Warunek końcowy:

Zakończenie przeglądania poprzez naciśnięcie przycisku.

16.4. Realizacja:

- **KIEROWNIK** lub **KLIENT** wybiera opcję DANE PRACOWNIKÓW
- Wyszukanie pracowników lub pracownika na podstawie podanych filtrów
- Wyświetlenie danych
- Zakończenie przeglądania przyciskiem POWRÓT

**KIEROWNIK** z tego poziomu ma możliwość zmodyfikowania danych wybranego pracownika.

**17. Przeglądanie statystyk zysków:**

17.1. Cel:

Uzyskanie informacji o najlepiej sprzedających się usługach.

17.2. Warunek początkowy:

Baza danych zawiera informacje o zamówieniach.

17.3. Warunek końcowy:

Zakończenie przeglądania poprzez naciśnięcie przycisku.

17.4. Realizacja:

- **WŁAŚCICIEL** wybiera opcję STATYSTYKI
- Wygenerowanie dokumentu zwierającego statystyki sprzedaży
- Zakończenie naciśnięciem przycisku POWRÓT

**18. Tworzenie kopii zapasowych:**

18.1. Cel:

Archiwizacja informacji, aby zapobiec utracie danych z bazy.

18.2. Warunek początkowy:

Baza danych zawiera dane.

18.3. Warunek końcowy:

Zakończenie archiwizacji poprzez utworzenie pliku, z którego można przywrócić bazę danych.

18.4. Realizacja:

- **ADMINISTRATOR** wybiera opcję UTWÓRZ KOPIĘ
- Zatwierdzenie operacji przyciskiem ZATWIERDŹ
- Utworzenie kopii

**19. Dodawanie konta właściciela:**

19.1. Cel:

Dodanie do bazy danych kont pracowników z domyślnymi uprawnieniami właściciela.

19.2. Warunek początkowy:

Istnieje kontro administratora.

19.3. Warunek końcowy:

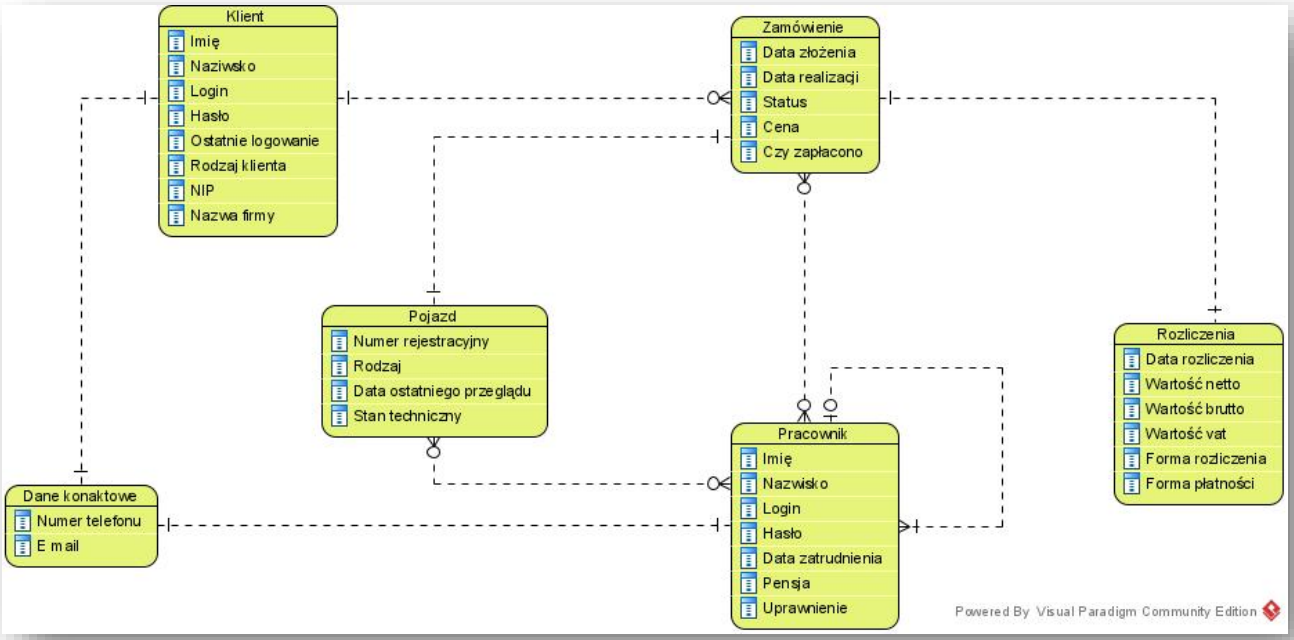
Wypełnienie formularza i zaakceptowanie operacji.

19.4. Realizacja:

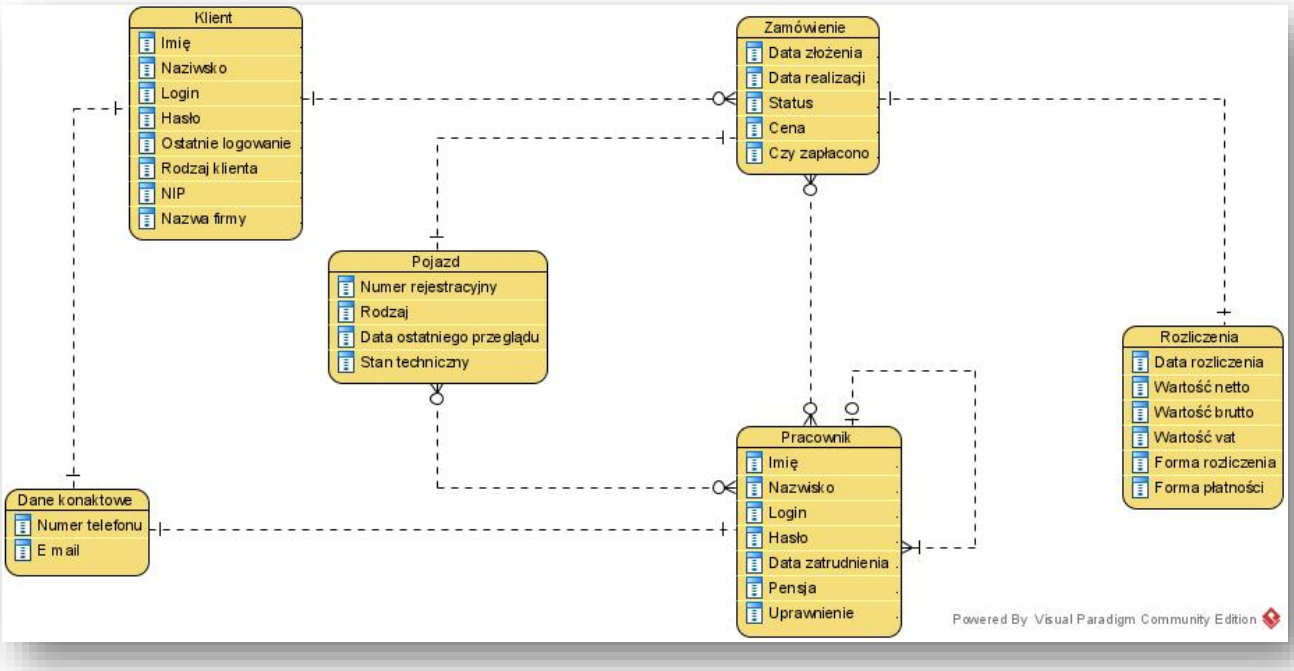
- **ADMINISTRATOR** wybiera opcję DODAJ WŁAŚCICIELA
- Wypełnienie formularza i weryfikacja prawdziwości danych
- Umieszczenie w bazie danych rekordu z danymi pracownika

## 1.4 Identyfikacja związków encji na podstawie scenariusza przypadków użycia

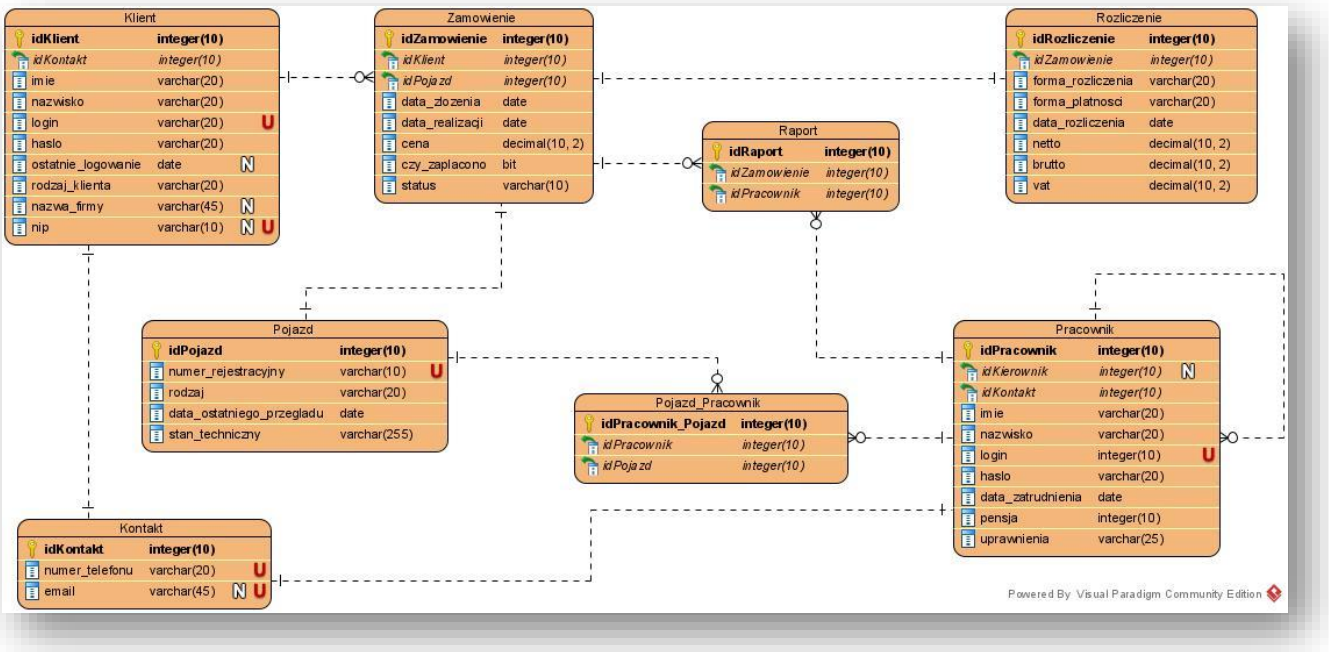
### 1.4.1 Diagramy związków encji




**Rysunek 2: Diagram konceptualny**




**Rysunek 3 : Diagram logiczny**




**Rysunek 4 : Diagram fizyczny**

### 1.4.2 Określenie tabel

- **Klient** (idKlient, *idKontakt*, imie, nazwisko, login, haslo, ostatnie_logowanie, rodzaj_klienta, nazwa_firmy, nip)

W tej tabeli przechowywane będą wszystkie niezbędne dane klientów zakładu oraz informacje o aktywności ich kont internetowych. Do jednego klienta może być przypisany tylko jeden rekord z tabeli **Kontakt.**

- **Pracownik** (idPracownik, *idKierownik*, *idKontakt*, imie, nazwisko, login, haslo, data_zatrudnienia, pensja, uprawnienia)

W tej tabeli przechowywane będą wszystkie niezbędne dane pracowników zakładu. Oprócz danych kontaktowych w tabeli znajdziemy datę zatrudnienia pracownika, jego uprawnienia oraz wysokość wynagrodzenia. Do jednego pracownika może być przypisany tylko jeden rekord z tabeli **Kontakt.** Pracownicy mają przypisaną osobę, która jest za nich odpowiedzialna (*idKierownik*).

- **Kontakt** *(*idKontakt, numer_telefonu, email)

Tabela ta przechowuje dane kontaktowe zarówno do pracowników jak i klientów. Nie wszystkie komórki muszą być wypełnione, obowiązkowe jest tylko podanie numeru telefonu, jednak wartości w tej tabeli są unikalne i nie mogą się powtórzyć.

- **Pojazd** (idPojazd, numer_rejestracyjny, rodzaj, data_ostatniego_przegladu, stan_techniczny)

Tutaj znajdą się informacje na temat pojazdów, którymi dysponuje firma. Możemy sprawdzić stan techniczny pojazdu, datę ostatniego przeglądu i rodzaj pojazdu.

- **Zamowienie** (idZamowienie, *idKlient, idPojazd,* data_zlozenia, data_realizacji, cena, czy_zaplacono, status)

W tej tabeli przechowujemy informacje na temat zamówienia składanego przez klienta. Możemy sprawdzić, który pojazd został zarezerwowany do zrealizowania zamówienia, kiedy zostało ono złożone, jaki ma status, kiedy było lub będzie zrealizowane, czy zostało opłacone i ile klient powinien zapłacić lub ile zapłacił. Jeden klient może złożyć wiele zamówień. Do jednego zamówienia może być wykorzystany tylko jeden pojazd. Jeżeli klient potrzebuje więcej pojazdów, musi złożyć osobne zamówienie.

- **Rozliczenie** (idRozliczenie, *idZamowienie,* forma_rozliczenia, forma_platnosci, data_rozliczenia, brutto, netto, vat)

Znajdują się tutaj zamówienia, które zostały już opłacone. Tabela pozwala generować raporty dochodów, ponieważ są w niej zawarte wszystkie informacje na temat rozliczeń. Każde rozliczenie dotyczy pojedynczego zamówienia. Forma rozliczenia i forma płatności może być taka sama dla wielu zamówień. Klient może poprosić o paragon lub fakturę na firmę, zapłacić kartą, gotówką lub przelewem.

- **Raport** (idRaport, *idZamowienie, idPracownik*)

Tabela ta rozwiązuje relacje wiele do wielu. Jeden pracownik może zrealizować kilka zamówień, ale też jedno zamówienie może być realizowane przez kilku pracowników.

- **Pracownik_Pojazd** (idPracownik_Pojazd, *idPracownik, idPojazd*)

Ta tabela również rozwiązuje relacje wiele do wielu, gdyż jeden pracownik może prowadzić kilka pojazdów. Na przykład mamy w firmie 3 busy i dany pracownik ma uprawnienia do prowadzenia każdego z tych pojazdów. Ale również jeden pojazd może być prowadzony przez kilku pracowników.

### 1.4.3 Określenie widoków

- Widok **Zamowienie** (*idZamowienie, idKlient, idPojazd*, imie, nazwisko, data_zlozenia, data_realizacji, rodzaj)
- Widok **Pracownik** (*idPracownik, idKontakt*, imie, nazwisko, uprawnienie, pensja, numer_telefonu)
- Widok **Klient** (*idKlient, idKontakt*, imie, nazwisko, rodzaj_klienta, numer_telefonu)
- Widok **Zamowienie_zrealizowane** (*idRozliczenie, idZamowienie, idKlient,* imie, nazwisko, data_zlozenia, data_rozliczenia, brutto, netto)

# 2 Projekt bazy danych

## 2.1 Analiza liczby instancji każdej encji

W ramach analizy liczby instancji każdej encji należy oszacować maksymalną liczbę danych, czyli maksymalny rozmiar tabeli, co pozwala oszacować rozmiar pamięci potrzebnej do przetrzymywania danych. Taka analiza zakłada, że dane zbierane są co najwyżej przez rok. Dane, które mają znaczenia głównie archiwalne, jak na przykład zamówienia, są archiwizowane. Archiwizacja nie dotyczy m.in. pracowników, klientów czy pojazdów.

**Tabela 1 : Analiza liczby instancji każdej encji**

| Tabela | Maksymalna liczba wierszy | Średnia liczba wierszy | Ilość kolumn | Maksymalny rozmiar tabeli | Średni rozmiar tabeli |
|---|---|---|---|---|---|
| Klient | 2000 | 1000 | 9 | 18000 | 9000 |
| Pracownik | 20 | 15 | 10 | 200 | 150 |
| Kontakt | 2020 | 1015 | 4 | 8080 | 4060 |
| Pojazd | 30 | 20 | 6 | 180 | 120 |
| Zamowienie | 5000 | 1500 | 7 | 35000 | 10500 |
| Rozliczenie | 5000 | 1500 | 8 | 35000 | 10500 |
| Raport | 10000 | 6000 | 3 | 30000 | 18000 |
| Pracownik_Pojazd | 600 | 300 | 3 | 1800 | 900 |

## 2.2 Analiza użycia identyfikująca podstawowe rodzaje transakcji

Analiza użycia ma na celu identyfikację podstawowych rodzajów transakcji, czyli dodawania, usuwania, modyfikacji i wyszukiwana oraz określenie na tej podstawie zmienności zawartości poszczególnych tabel. Dla każdej tabeli należy określić jaki rodzaj transakcji będzie przeprowadzany najczęściej. Nie wyklucza to pozostałych operacji na tabelach, na wszystkich tabelach można przeprowadzać wszystkie podstawowe transakcje, natomiast transakcje wymienione poniżej są operacjami wykonywanymi najczęściej.

**Tabela 2 : Analiza użycia identyfikująca podstawowe rodzaje transakcji**

| Tabela | Rodzaj transakcji |
|---|---|
| Klient | wyszukiwanie, wstawianie *(imie, nazwisko)* |
| Pracownik | wyszukiwanie, wstawianie *(imie, nazwisko)* |
| Kontakt | wyszukiwanie, wstawianie, modyfikacja *(numer_telefonu)* |
| Pojazd | wyszukiwanie *(numer_rejestracyjny)* |
| Zamowienie | wstawianie, wyszukiwanie, modyfikacja *(idKlient, data_zlozenia)* |
| Rozliczenie | wstawianie, wyszukiwanie *(data_rozliczenia, netto)* |

## 2.3 Sformułowanie wymagań dotyczących dostępu – określenie częstości wykonywania operacji na danych

**Tabela 3 : Określenie częstości wykonywania poszczególnych operacji na danych**

| L.p. | Operacja | Częstotliwość wykonywania | Maksymalny czas wykonania zapytania |
|---|---|---|---|
| 1 | Roczny raport przychodów | Raz na rok | 30 min |
| 2 | Roczny raport podatku VAT | Raz na rok | 30 minut |
| 3 | Miesięczny raport płac dla pracowników | Raz na miesiąc | 5 minut |
| 4 | Miesięczny raport przychodów | Raz na miesiąc | 5 minut |
| 5 | Dzienny raport przychodów | Raz na dzień | 2 minuty |
| 6 | Raport historii klienta – na żądanie | Do 500 razy dziennie | 5 sekund |
| 7 | Sprawdzenie statusu zamówienia | Do 1000 razy dziennie | 5 sekund |
| 8 | Dodanie klienta | Do 300 razy dziennie | 5 sekund |
| 9 | Dodanie pracownika | Do 10 razy rocznie | 5 sekund |
| 10 | Zmiana statusu zamówienia | Do 500 razy dziennie | 5 sekund |
| 11 | Tworzenie kopii zapasowej bazy danych | Raz na tydzień | 10 godzin |
| 12 | Wczytywanie kopii bazy z pliku | Raz w roku | 10 godzin |

Opis przebiegów operacji:

1. Wygenerowanie rocznego raportu przychodów wymaga dostępu do tabeli **Zamowienie** w celu uzyskania informacji o wszystkich zamówieniach w roku oraz tabeli **Rozliczenie**.
2. Operacja generująca roczny raport podatku VAT wymaga dostępu do tabeli **Rozliczenie**.
3. Miesięczny raport płac dla pracowników to operacja korzystająca z tabeli **Pracownik**.
4. Wygenerowanie miesięcznego raportu przychodów wymaga dostępu do tabeli **Zamowienie** w celu uzyskania informacji o wszystkich zamówieniach w miesiącu oraz tabeli **Rozliczenie**.
5. Wygenerowanie dziennego raportu przychodów wymaga dostępu do tabeli **Zamowienie** w celu uzyskania informacji o wszystkich zamówieniach w ciągu dnia oraz tabeli **Rozliczenie**.
6. Raport historii klienta to operacja wykonywana na żądanie i polega na uzyskaniu wszystkich zapamiętanych wizyt danego klienta. Potrzebny jest dostęp do tabeli **Klient** oraz **Zamowienie**, ewentualnie **Rozliczenie**.
7. Sprawdzenie statusu zamówienia wymaga dostępu do tabeli **Zamowienie**.
8. Dodanie klienta to operacja dodająca rekord do tabeli **Klient**, wiąże się ona z dodaniem nowego rekordu w tabeli **Kontakt.**
9. Dodanie pracownika to operacja dodająca rekord do tabeli **Pracownik**, wiąże się z dodanie nowego rekordu w tabeli **Kontakt.**
10. Zmiana statusu zamówienia przebiega identycznie jak sprawdzenie statusu, wymaga dostępu do tabeli **Zamowienie**.
11. Tworzenie kopii zapasowej bazy danych jest wykonywanie cyklicznie w odstępie tygodnia i ma na celu zapewnienie bezpieczeństwa danych. Odbywa się ono w nocy, gdy nie dochodzi do modyfikacji zawartości bazy. Operacja wymaga dostępu do wszystkich tabel.
12. Wczytywanie kopii bazy z pliku jest podobne do tworzenia kopii – wymaga dostępu do wszystkich tabel i jest wykonywane w nocy, gdy firma jest zamknięta.

## 2.4 Analiza integralności

System bazodanowy powinien zawierać odpowiednie mechanizmy zabezpieczające przed skutkami przypadkowych błędów logicznych, konfliktów we współbieżnym dostępie do danych oraz skutkami awarii oprogramowania i sprzętu komputerowego. Ponadto system powinien zawsze dostarczać wiarygodne dane i być zabezpieczony przed nieautoryzowaną modyfikacją informacji. System baz danych powinien też zapewniać możliwość sprawdzania i ewentualnej korekty wprowadzanych danych oraz powinny zawierać odpowiednie mechanizmy zapewniające prawidłowe przetwarzanie danych. Konieczne jest także zapewnienie kompletności, poprawności i wiarygodności danych zgromadzonych w bazie. Proces ochrony integralności obejmuje:

- kontrolę danych wejściowych oraz synchronizację dostępu do danych
- poprawianie czyli korektę danych, cofanie i odtwarzanie stanu bazy
- archiwizacje poprzez tworzenie kopii bazy oraz zapisów działania systemu
- testowanie czyli sprawdzanie poprawności zawartości bazy

Integralność dotyczy poprawnie zaprojektowanego schematu bazy danych oraz spełnienia ograniczeń nałożonych na wartości atrybutów opisujących obiekty w bazie. Zakładamy trzy typy więzów integralności wewnętrznej:

- klucza głównego (np. *idKlient* w tabeli **Klient** nie może być równy null)
- klucza obcego (np. *idKontakt* w tabeli **Klient** nie może być równy null)
- dziedziny wartości np. atrybut *forma_platnosci* w tabeli **Rozliczenie** może przybierać tylko następujące wartości: *karta płatnicza, gotówka*, ewentualnie *przelew.*

Zakładamy także następujące typy więzów integralności dodatkowej:

- więzy przejścia: np. jeśli usuwamy dane na temat pracownika z bazy, musimy również usunąć odpowiedni wiersz w tabeli *Kontakt*
- więzy statyczne: np. liczba rezerwacji danego pojazdu nie może przekroczyć ilości dostępnych pojazdów w zakładzie
- więzy atomowości transakcji: np. przy rezerwacji terminu wizyty nie można dopuścić do spowodowania anomalii wynikających z opóźnienia w komunikacji aplikacja-baza danych oraz anomalii wynikających z próby rejestracji w tym samym terminie przez więcej niż jedną osobę
- więzy ochrony danych: np. z poziomu użytkownika aplikacji niemożliwe jest usunięcie danych o pracownikach

## 2.5 Dostrajanie bazy danych pod względem wydajności

### 2.5.1 Tworzenie mechanizmów dostępu związanych z przechowywaniem

Dzięki analizie liczby instancji oraz użycia, można zastosować mechanizmy poprawiające wydajność bazy danych. Do takich mechanizmów zalicza się sekwencyjność, haszowanie oraz klastry.

#### Sekwencyjność

Sekwencyjność można zastosować w przypadku plików uporządkowanych rekordów. Polega to na porządkowaniu rekordów pliku według wartości jednego pola. Rekordy można w ten sposób uporządkować np. alfabetycznie lub jeśli wartość pola jest formatu daty, od najstarszego do najnowszego. Odczyt takich rekordów jest wydajny i nie wymaga sortowania. Do wyszukiwania może być użyty efektywny mechanizm wyszukiwania binarnego. Niestety wstawianie rekordu jest kosztowne, gdyż wymaga fizycznego uporządkowania rekordów, zatem należy wyszukać pozycję i przesuwać średnio połowę rekordów. Kosztowną operacją jest również usuwanie, gdyż podobnie jak w przypadku dodania, wymaga reorganizacji. Wyszukanie rekordu wg. innego klucza niż klucz uporządkowania jest liniowe. Modyfikacja zależy od warunków wyszukiwania i modyfikowanego pola. Na podstawie tej analizy, sekwencyjność powinna zostać zastosowania w tabelach **Pojazd, Pracownik** oraz **Klient**.

#### Klastry

Zapis kilku tabel w klaster sprawia, że ich złączenie staje się szybsze, ale opracje na pojedynczych tabelach są nieznacznie wolniejsze. Częstym złączeniem będzie wyszukanie klienta oraz jego danych kontaktowych. Tabele zawierające te dane powinny znaleźć się w jednym klastrze, podobnie jak tabele **Zamowienie** i **Rozliczenie**

### 2.5.2 Dodawanie indeksów

Przy analizie dodawania do bazy indeksów trzeba mieć na uwadze czas wykonywania zapytań oraz czas modyfikowania, wstawiania i usuwania danych. Indeks jest strukturą danych zwiększającą szybkość wykonywania operacji wyszukiwania na tabeli. Trzeba zatem zlokalizować pola po który najczęściej dokonywane jest wyszukiwanie:

- **Klient** - *id_klient*, imie, nazwisko
- **Pracownik** - *id_pracownik*, imie, nazwisko
- **Pojazd** - *id_pojazd*, numer_rejestracyjny
- **Zamowienie** - *id_zamowienie*, id_klient, id_status, data_zlozenia
- **Rozliczenie** - *id_rozliczenie*, id_zamowienie, data_rozliczenia
- **Kontakt** – *id_dane_kontaktowe*, numer_telefonu

Niestety użycie indeksów wpływa negatywnie na czas modyfikowania, wstawiania i usuwania danych. Oprócz podstawowych indeksów stworzonych na podstawie id, zdecydowałyśmy stworzyć indeksy w tabelach **Klient** i **Pracownik** na podstawie atrybutów imie i nazwisko.

### 2.5.3 Denormalizacja

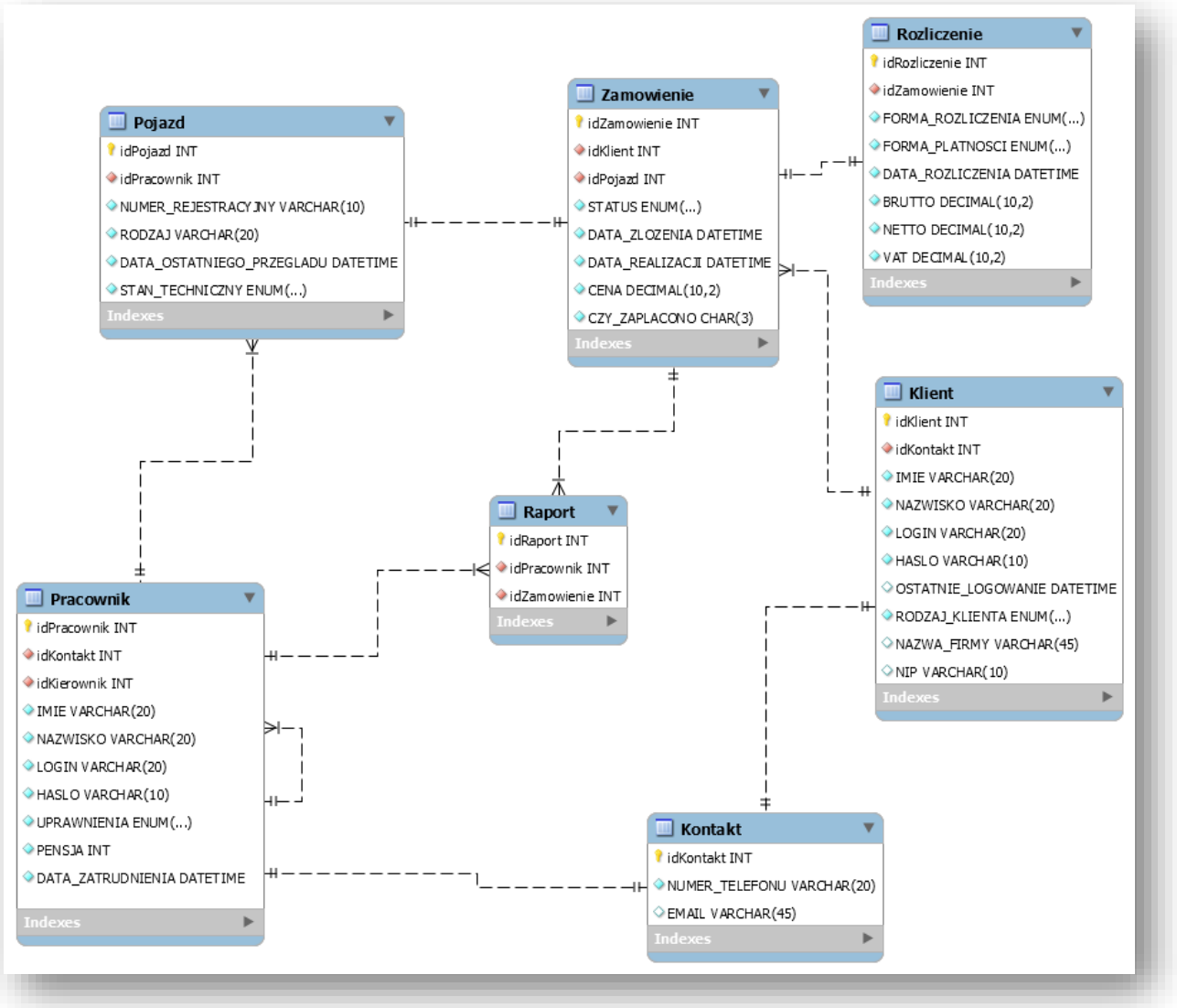
Denormalizacja polega na wprowadzeniu kontrolowanej nadmierności do bazy danych w celu przyspieszenia wykonywania na niej operacji. Dzięki denormalizacji bazy danych unika się kosztownych operacji połączenia tabel.

Wstępna denormalizacja została przeprowadzona na etapie tworzenia modelu konceptualnego bazy danych. Pierwotnie chciałyśmy połączyć tabele pracownik i klient, jednak ostatecznie zrezygnowałyśmy z takiego połączenia ze względu na wiele pustych komórek i możliwości pojawienia się danych nie w tych miejscach. Dane kontaktowe wszystkich użytkowników trzymamy w osobnej tabeli. Na początku planowałyśmy stworzyć tabelę z wynagrodzeniem pracowników w zależności od uprawnień, ale finalnie pensja zawarta jest w tabeli pracownik. Wcześniej stworzyłyśmy kilka tabel, w których przechowywanych było stosunkowo niewiele rekordów np. uprawnienie, status w celu przechowywania tam wartości numerycznych, jednak system udostępnił nam możliwość ustawienia atrybutu w tabeli na wartość numeryczną.

### 2.5.4 Wykorzystanie możliwości wybranego systemu zarządzania bazą danych

Ważny jest wybór Systemu Zarządzania Bazą Danych ze względu na wydajność bazy. SZBD różnią się typem optymalizacji, stopniem wielowątkowości, rodzajem zapytań itd.

Do stworzenia naszej bazy danych wybrałyśmy MySQL, ponieważ posiada liczne narzędzia służące zwiększeniu wydajności. Początkowo chciałyśmy stworzyć tabele, gdzie przechowywane będą jedynie wartości enumeryczne, jednak system umożliwił nam pozbycie się takich tabel i stworzenie w docelowych tabelach kolumn, które takie wartości mogą przechowywać. Podczas tworzenia diagramu w Visual Paradigm nie miałyśmy takiej opcji. MySQL umożliwił również stworzenie końcowego diagramu, który w pełni odpowiada wartościom przechowywanym w naszej bazie. MySQL umożliwia również sprawne posługiwanie się wszystkimi znakami z języka polskiego.



### 2.5.5 Analiza bezpieczeństwa i poufności

Z bazy danych korzystają pracownicy zakładu, każdy na poziomie swoich uprawnień.

- **Administrator** ma dostęp do wszystkich tabeli w bazie ze względu na obowiązek tworzenia kopii zapasowych. Jest on potrzebny tylko od strony technicznej, nie musi mieć swojego dostępu od strony aplikacji.
- **Właściciel** również ma dostęp do wszystkich tabeli, ponieważ ma on wgląd do danych pracowników i jednocześnie pełni funkcje kierowników.
- **Kierownik** ma dostęp do wszystkich tabeli z wyjątkiem informacji o danych innych pracowników.
- W zakładzie pracują również kierowcy i mechanicy. Mają oni dostęp jedynie do tabeli pojazd w celu sprawdzenia np. stanu technicznego lub daty ostatniego przeglądu. Ze względu na to, że mają takie same uprawnienia, będą nosili nazwę **pracownik**.

# 3 Implementacja bazy danych

## 3.1 Tworzenie tabel

```
-- -----------------------------------------------------
-- Table `zakład_transportowy`.`Pojazd`
-- -----------------------------------------------------

DROP TABLE IF EXISTS `zakład_transportowy`.`Pojazd`;
CREATE TABLE IF NOT EXISTS `zakład_transportowy`.`Pojazd` (
   `idPojazd` INT NOT NULL AUTO_INCREMENT,
   `NUMER_REJESTRACYJNY` VARCHAR(10) CHARACTER SET 'utf8' COLLATE 'utf8_polish_ci' NOT NULL UNIQUE,
   `RODZAJ` ENUM('BUS', 'TIR' ,'CIĘŻARÓWKA', 'CYSTERNA', 'OSOBÓWKA') CHARACTER SET 'utf8'COLLATE
        'utf8_polish_ci' NOT NULL,
   `DATA_OSTATNIEGO_PRZEGLADU` DATETIME NOT NULL,
   `STAN_TECHNICZNY` ENUM('SPRAWNY', 'POTRZEBNY PRZEGLAD', 'W NAPRAWIE', 'DO NAPRAWY') CHARACTER SET
        'utf8' COLLATE 'utf8_polish_ci' NOT NULL,
   PRIMARY KEY (`idPojazd`),
   UNIQUE INDEX `NUMER_REJESTRACYJNY_UNIQUE` (`NUMER_REJESTRACYJNY` ASC))
ENGINE = InnoDB
DEFAULT CHARACTER SET = utf8
COLLATE = utf8_polish_ci;



-- -----------------------------------------------------
-- Table `zakład_transportowy`.`Kontakt`
-- -----------------------------------------------------

DROP TABLE IF EXISTS `zakład_transportowy`.`Kontakt`;
CREATE TABLE IF NOT EXISTS `zakład_transportowy`.`Kontakt` (
   `idKontakt` INT NOT NULL AUTO_INCREMENT,
   `NUMER_TELEFONU` VARCHAR(20) CHARACTER SET 'utf8' COLLATE 'utf8_polish_ci' NOT NULL UNIQUE,
   `EMAIL` VARCHAR(45) CHARACTER SET 'utf8' COLLATE 'utf8_polish_ci' NULL UNIQUE,
   PRIMARY KEY (`idKontakt`),
   UNIQUE INDEX `NUMER_TELEFONU_UNIQUE` (`NUMER_TELEFONU` ASC),
   UNIQUE INDEX `EMAIL_UNIQUE` (`EMAIL` ASC))
ENGINE = InnoDB
DEFAULT CHARACTER SET = utf8
COLLATE = utf8_polish_ci;
```

```
-- -----------------------------------------------------
-- Table `zakład_transportowy`.`Klient`
-- -----------------------------------------------------

DROP TABLE IF EXISTS `zakład_transportowy`.`Klient`;
CREATE TABLE IF NOT EXISTS `zakład_transportowy`.`Klient` (
   `idKlient` INT NOT NULL AUTO_INCREMENT,
   `idKontakt` INT NOT NULL,
   `IMIE` VARCHAR(20) CHARACTER SET 'utf8' COLLATE 'utf8_polish_ci' NOT NULL,
   `NAZWISKO` VARCHAR(20) CHARACTER SET 'utf8' COLLATE 'utf8_polish_ci' NOT NULL,
   `LOGIN` VARCHAR(20) CHARACTER SET 'utf8' COLLATE 'utf8_polish_ci' NOT NULL UNIQUE,
   `HASLO` VARCHAR(20) CHARACTER SET 'utf8' COLLATE 'utf8_polish_ci' NOT NULL,
   `OSTATNIE_LOGOWANIE` DATETIME NULL,
   `RODZAJ_KLIENTA` ENUM('OSOBA FIZYCZNA', 'FIRMA') CHARACTER SET 'utf8' COLLATE 'utf8_polish_ci' NOT
           NULL,
   `NAZWA_FIRMY` VARCHAR(45) CHARACTER SET 'utf8' COLLATE 'utf8_polish_ci' NULL UNIQUE,
   `NIP` VARCHAR(10) CHARACTER SET 'utf8' COLLATE 'utf8_polish_ci' NULL UNIQUE,
   PRIMARY KEY (`idKlient`),
   INDEX `fk_Klient_Kontakt1_idx` (`idKontakt` ASC),
   UNIQUE INDEX `NIP_UNIQUE` (`NIP` ASC),
   CONSTRAINT `fk_Klient_Kontakt1`
      FOREIGN KEY (`idKontakt`)
      REFERENCES `zakład_transportowy`.`Kontakt` (`idKontakt`)
      ON DELETE CASCADE
      ON UPDATE CASCADE)
ENGINE = InnoDB
DEFAULT CHARACTER SET = utf8
COLLATE = utf8_polish_ci;
```

```
-- -----------------------------------------------------
-- Table `zakład_transportowy`.`Pracownik`
-- -----------------------------------------------------

DROP TABLE IF EXISTS `zakład_transportowy`.`Klient`;
CREATE TABLE IF NOT EXISTS `zakład_transportowy`.`Klient` (
   `idKlient` INT NOT NULL AUTO_INCREMENT,
   `idKontakt` INT NOT NULL,
   `idKierownik` INT,
   `IMIE` VARCHAR(20) CHARACTER SET 'utf8' COLLATE 'utf8_polish_ci' NOT NULL,
   `NAZWISKO` VARCHAR(20) CHARACTER SET 'utf8' COLLATE 'utf8_polish_ci' NOT NULL,
   `LOGIN` VARCHAR(20) CHARACTER SET 'utf8' COLLATE 'utf8_polish_ci' NOT NULL UNIQUE,
   `HASLO` VARCHAR(20) CHARACTER SET 'utf8' COLLATE 'utf8_polish_ci' NOT NULL,
   `UPRAWNIENIA` ENUM('ADMINISTRATOR', 'WLASCICIEL', 'KIEROWNIK', 'MECHANIK', 'KIEROWCA')
        CHARACTER SET 'utf8' COLLATE 'utf8_polish_ci' NOT NULL,
   `PENSJA` INT NOT NULL,
   `DATA_ZATRUDNIENIA` DATETIME NOT NULL,
   PRIMARY KEY (`idPracownik`),
   INDEX `fk_Pracownik_Kontakt1_idx` (`idKontakt` ASC),
   INDEX `fk_Pracownik_Pracownik1_idx` (`idKierownik` ASC),
   UNIQUE INDEX `LOGIN_UNIQUE` (`LOGIN` ASC),
   UNIQUE INDEX `HASLO_UNIQUE` (`HASLO` ASC),
   CONSTRAINT `fk_Pracownik_Kontakt1`
      FOREIGN KEY (`idKontakt`)
      REFERENCES `zakład_transportowy`.`Kontakt` (`idKontakt`)
      ON DELETE CASCADE
      ON UPDATE CASCADE,
   CONSTRAINT `fk_Pracownik_Pracownik1`
      FOREIGN KEY (`idKierownik`)
      REFERENCES `zakład_transportowy`.`Pracownik` (`idPracownik`)
      ON DELETE NO ACTION
      ON UPDATE NO ACTION)
ENGINE = InnoDB
DEFAULT CHARACTER SET = utf8
COLLATE = utf8_polish_ci;
```

```sql
-- -----------------------------------------------------
-- Table `zakład_transportowy`.`Zamowienie`
-- -----------------------------------------------------

DROP TABLE IF EXISTS `zakład_transportowy`.`Zamowienie`;
CREATE TABLE IF NOT EXISTS `zakład_transportowy`.`Zamowienie` (
   `idZamowienie` INT NOT NULL AUTO_INCREMENT,
   `idKlient` INT,
   `idPojazd` INT NOT NULL,
   `STATUS` ENUM('ZLOZONO', 'W TRAKCIE', 'ZREALIZOWANO') CHARACTER SET 'utf8' COLLATE 'utf8_polish_ci'
           NOT NULL,
   `DATA_ZLOZENIA` DATE NOT NULL,
   `DATA_REALIZACJI` DATE NOT NULL,
   `CENA` DECIMAL(10,2) NOT NULL,
   `CZY_ZAPLACONO` CHAR(3) CHARACTER SET 'utf8' COLLATE 'utf8_polish_ci' NOT NULL,
   PRIMARY KEY (`idZamowienie`),
   INDEX `fk_Zamowienie_Klient1_idx` (`idKlient` ASC),
   INDEX `fk_Zamowienie_Pojazd1_idx` (`idPojazd` ASC),
   CONSTRAINT `fk_Zamowienie_Klient1`
      FOREIGN KEY (`idKlient`)
      REFERENCES `zakład_transportowy`.`Klient` (`idKlient`)
      ON DELETE NO ACTION
      ON UPDATE NO ACTION,
   CONSTRAINT `fk_Zamowienie_Pojazd1`
      FOREIGN KEY (`idPojazd`)
      REFERENCES `zakład_transportowy`.`Pojazd` (`idPojazd`)
      ON DELETE NO ACTION
      ON UPDATE NO ACTION)
ENGINE = InnoDB
DEFAULT CHARACTER SET = utf8
COLLATE = utf8_polish_ci;
```

```
-- -----------------------------------------------------
-- Table `zakład_transportowy`.`Rozliczenie`
-- -----------------------------------------------------
DROP TABLE IF EXISTS `zakład_transportowy`.`Rozliczenie`;
CREATE TABLE IF NOT EXISTS `zakład_transportowy`.`Rozliczenie` (
   `idRozliczenie` INT NOT NULL AUTO_INCREMENT,
   `idZamowienie` INT NOT NULL,
   `FORMA_ROZLICZENIA` ENUM('PARAGON', 'FAKTURA') CHARACTER SET 'utf8' COLLATE 'utf8_polish_ci' NOT NULL,
   `FORMA_PLATNOSCI` ENUM('PRZELEW', 'GOTOWKA', 'KARTA PLATNICZA') CHARACTER SET 'utf8' COLLATE
        'utf8_polish_ci' NOT NULL,
   `DATA_ROZLICZENIA` DATE NOT NULL,
   `BRUTTO` DECIMAL(10,2) NOT NULL,
   `NETTO` DECIMAL(10,2) NOT NULL,
   `VAT` DECIMAL(10,2) NOT NULL,
   PRIMARY KEY (`idRozliczenie`),
   INDEX `fk_Rozliczenie_Zamowienie1_idx` (`idZamowienie`ASC),
   CONSTRAINT `fk_Rozliczenie_Zamowienie1`
      FOREIGN KEY (`idZamowienie`)
      REFERENCES `zakład_transportowy`.`Zamowienie` (`idZamowienie`)
      ON DELETE NO ACTION
      ON UPDATE NO ACTION)
ENGINE = InnoDB
DEFAULT CHARACTER SET = utf8
COLLATE = utf8_polish_ci;
-- -----------------------------------------------------
-- Table `zakład_transportowy`.`Pracownik_Pojazd`
-- -----------------------------------------------------
DROP TABLE IF EXISTS `zakład_transportowy`.`Pracownik_Pojazd`;
CREATE TABLE IF NOT EXISTS `zakład_transportowy`.`Pracownik_Pojazd` (
   `idPracownik_Pojazd` INT NOT NULL,
   `idPojazd` INT NOT NULL,
   `idPracownik` INT NOT NULL,
   PRIMARY KEY (`idPracownik_Pojazd`),
   INDEX `fk_Pracownik_Pojazd_Pojazd1_idx` (`idPojazd`ASC),
   INDEX `fk_Pracownik_Pojazd_Pracownik1_idx` (`idPracownik` ASC),
   CONSTRAINT `fk_Pracownik_Pojazd_Pojazd1`
      FOREIGN KEY (`idPojazd`)
      REFERENCES `zakład_transportowy`.`Pojazd` (`idPojazd`)
         ON DELETE NO ACTION
         ON UPDATE NO ACTION,
   CONSTRAINT `fk_Pracownik_Pojazd_Pracownik1`
         FOREIGN KEY (`idPracownik`)
         REFERENCES `zakład_transportowy`.`Pracownik` (`idPracownik`)
         ON DELETE NO ACTION
         ON UPDATE NO ACTION)
ENGINE = InnoDB;
DEFAULT CHARACTER SET = utf8
COLLATE = utf8_polish_ci;
```

```
-- -----------------------------------------------------
-- Table `zakład_transportowy`.`Raport`
-- -----------------------------------------------------

DROP TABLE IF EXISTS `zakład_transportowy`.`Raport`;
CREATE TABLE IF NOT EXISTS `zakład_transportowy`.`Raport` (
   `idRaport` INT NOT NULL AUTO_INCREMENT,
   `idPracownik` INT NOT NULL,
   `idZamowienie` INT NOT NULL,
   PRIMARY KEY (`idRaport`),
   INDEX `fk_Raport_Pracownik1_idx` (`idPracownik` ASC),
   INDEX `fk_Raport_Zamowienie1_idx` (`idZamowienie` ASC),
   CONSTRAINT `fk_Raport_Pracownik1`
      FOREIGN KEY (`idPracownik`)
      REFERENCES `zakład_transportowy`.`Pracownik` (`idPracownik`)
      ON DELETE NO ACTION
      ON UPDATE NO ACTION,
   CONSTRAINT `fk_Raport_Zamowienie1`
      FOREIGN KEY (`idZamowienie`)
      REFERENCES `zakład_transportowy`.`Zamowienie` (`idZamowienie`)
      ON DELETE CASCADE
      ON UPDATE CASCADE)
ENGINE = InnoDB
DEFAULT CHARACTER SET = utf8
COLLATE = utf8_polish_ci;
```

## 3.2 Tworzenie widoków

```
-- -----------------------------------------------------
-- `Widok Zamowienie`
-- -----------------------------------------------------

DROP VIEW IF EXISTS `View_Zamowienie`//
CREATE VIEW `View_Zamowienie` AS
   SELECT `idZamowienie`, Zamowienie.idKlient, Zamowienie.idPojazd, Klient.IMIE, Klient.NAZWISKO,
            `DATA_ZLOZENIA`,`DATA_REALIZACJI`, Pojazd.Rodzaj
   FROM `Zamowienie`
   JOIN Klient ON Zamowienie.idZamowienie = Klient.idKlient
   JOIN Pojazd ON Zamowienie.idPojazd = Pojazd.idPojazd
   //


-- -----------------------------------------------------
-- `Widok Pracownik`
-- -----------------------------------------------------

DROP VIEW IF EXISTS `View_Pracownik`//
CREATE VIEW `View_Pracownik` AS
   SELECT `idPracownik`, Pracownik.idKontakt, `IMIE`, `NAZWISKO`, `UPRAWNIENIA`, `PENSJA`,
            Kontakt.NUMER_TELEFONU
   FROM `Pracownik`
   JOIN Kontakt ON Pracownik.idKontakt = Kontakt.idKontakt
   //


-- -----------------------------------------------------
-- `Widok Klient`
-- -----------------------------------------------------

DROP VIEW IF EXISTS `View_Klient`//
CREATE VIEW `View_Klient` AS
   SELECT `idKlient`, Klient.idKontakt, `IMIE`, `NAZWISKO`, `Rodzaj_KLIENTA`, Kontakt.NUMER_TELEFONU
   FROM `Klient`
   JOIN Kontakt ON Klient.idKontakt = Kontakt.idKontakt
   //



-- -----------------------------------------------------
-- `Widok Zamowienie_zrealizowane`
-- -----------------------------------------------------

DROP VIEW IF EXISTS `View_Zamowienie_zrealizowane`//
CREATE VIEW `View_Zamowienie_zrealizowane` AS
   SELECT `idRozliczenie`, Rozliczenie.idZamowienie, Zamowienie.idKlient, Klient.IMIE, KLIENT.NAZWISKO,
            Zamowienie.DATA_ZLOZENIA, `DATA_ROZLICZENIA`, `BRUTTO`,`NETTO`
   FROM `Rozliczenie`
   JOIN Zamowienie ON Rozliczenie.idZamowienie = Zamowienie.idZamowienie
   JOIN Klient ON Zamowienie.idKlient = Klient.idKlient
   //
```

## 3.3 Tworzenie wyzwalaczy

```
-- -----------------------------------------------------
-- `Wyzwalacz usuniecie_konta_klienta`
-- -----------------------------------------------------

DROP TRIGGER IF EXISTS `usuniecie_konta_klienta`//
CREATE TRIGGER  `usuniecie_konta_klienta`
BEFORE DELETE ON `Klient`
FOR EACH ROW
   BEGIN
   DELETE FROM `Kontakt` WHERE idKontakt = old.idKontakt;
   UPDATE `Zamowienie` SET idKlient = NULL WHERE idKlient = old.idKlient;
   END
   //



-- -----------------------------------------------------
-- `Wyzwalacz usuniecie_konta_pracownika`
-- -----------------------------------------------------

DROP TRIGGER IF EXISTS `usuniecie_konta_pracownika`//
CREATE TRIGGER `usuniecie_konta_pracownika`
BEFORE DELETE ON `Pracownik`
FOR EACH ROW
   BEGIN
   DELETE FROM `Pracownik_Pojazd` WHERE idPracownik = old.idPracownik;
   DELETE FROM `Kontakt` WHERE idkontakt = old.idKontakt;
   UPDATE `Raport` SET idPracownik = NULL WHERE idPracownik = old.idPracownik;
   END
   //



-- -----------------------------------------------------
-- `Wyzwalacz usuniecie_pojazdu`
-- -----------------------------------------------------

DROP TRIGGER IF EXISTS `usuniecie_pojazdu`//
CREATE TRIGGER `usuniecie_pojazdu`
BEFORE DELETE ON `Pojazd`
FOR EACH ROW
   BEGIN
   DELETE FROM `Pracownik_Pojazd` WHERE idPojazd = old.idPojazd;
   END
   //
```

## 3.4 Tworzenie funkcji

```
-- -----------------------------------------------------
-- `Funkcja dochod`
-- -----------------------------------------------------

DROP FUNCTION IF EXISTS `dochod`;
delimiter //
CREATE FUNCTION dochod() RETURNS DECIMAL(10,2)
   BEGIN
   RETURN (SELECT SUM(NETTO) FROM Rozliczenie);
   END
   //


-- -----------------------------------------------------
-- `Funkcja dochod_dnia`
-- -----------------------------------------------------

DROP FUNCTION IF EXISTS `dochod_dnia`;
delimiter //
CREATE FUNCTION dochod_dnia(dzien DATE) RETURNS DECIMAL(10,2)
   BEGIN
   RETURN (SELECT SUM(NETTO) FROM Rozliczenie WHERE DATA_ROZLICZENIA = dzien);
   END
   //


-- -----------------------------------------------------
-- `Funkcja dochod_z_okresu_czasu`
-- -----------------------------------------------------

DROP FUNCTION IF EXISTS `dochod_z_okresu_czasu`;
delimiter //
CREATE FUNCTION dochod_z_okresu_czasu(odDnia DATE, doDnia DATE) RETURNS DECIMAL(10,2)
   BEGIN
   RETURN (SELECT SUM(NETTO) FROM Rozliczenie WHERE DATA_ROZLICZENIA >= odDnia and
            DATA_ROZLICZENIA <= doDnia);
   END
   //


-- -----------------------------------------------------
-- `Funkcja srednie_zarobki`
-- -----------------------------------------------------

DROP FUNCTION IF EXISTS `srednie_zarobki`;
delimiter //
CREATE FUNCTION srednie_zarobki() RETURNS INT
   BEGIN
   DECLARE zarobki INT;
   SET zarobki = (SELECT AVG(pensja) FROM Pracownik);
   RETURN zarobki;
   END
    //
```

```
-- -----------------------------------------------------
-- `Funkcja vat`
-- -----------------------------------------------------

DROP FUNCTION IF EXISTS `vat`;
delimiter //
CREATE FUNCTION vat(cena DECIMAL(10,2)) RETURNS DECIMAL(10,2)
   BEGIN
   RETURN 0.23 * cena;
   END
    //


-- -----------------------------------------------------
-- `Funkcja vat_z_okresu_czasu`
-- -----------------------------------------------------

DROP FUNCTION IF EXISTS `vat_z_okresu_czasu`;
delimiter //
CREATE FUNCTION vat_z_okresu_czasu(odDnia DATE, doDnia DATE) RETURNS DECIMAL(10,2)
   BEGIN
   RETURN (SELECT SUM(VAT) FROM Rozliczenie WHERE DATA_ROZLICZENIA >= odDnia and
         DATA_ROZLICZENIA <= doDnia);
   END
    //


-- -----------------------------------------------------
-- `Funkcja netto`
-- -----------------------------------------------------

DROP FUNCTION IF EXISTS `netto`;
delimiter //
CREATE FUNCTION netto(cena DECIMAL(10,2)) RETURNS DECIMAL(10,2)
   BEGIN
   RETURN 0.77 * cena;
   END
    //
```

## 3.5 Tworzenie indeksów

```
-- -----------------------------------------------------
-- `Indeksy`
-- -----------------------------------------------------

CREATE INDEX imie_nazwisko ON Klient(IMIE, NAZWISKO);
CREATE INDEX imie_nazwisko ON Pracownik(IMIE, NAZWISKO);
```

## 3.6 Tworzenie użytkowników

```
-- -----------------------------------------------------
-- `Uzytkownicy`
-- -----------------------------------------------------

DROP USER IF EXISTS Klient;
flush privileges;
CREATE USER Klient IDENTIFIED BY 'kLient12';
DROP USER IF EXISTS Właściciel;
flush privileges;
CREATE USER Właściciel IDENTIFIED BY 'sZef12';
DROP USER IF EXISTS Kierownik;
flush privileges;
CREATE USER Kierownik IDENTIFIED BY 'kIerownik12';
DROP USER IF EXISTS Pracownik;
flush privileges;
CREATE USER Pracownik IDENTIFIED BY 'pRacownik12';

FLUSH PRIVILEGES;
GRANT SELECT, INSERT ON `zakład_transportowy`.Zamowienie TO Klient;
GRANT SELECT, DELETE ON `zakład_transportowy`.Klient TO Klient;
GRANT SELECT, UPDATE ON `zakład_transportowy`.Kontakt TO Klient;

GRANT SELECT ON `zakład_transportowy`.Zamowienie TO Pracownik;
GRANT SELECT ON `zakład_transportowy`.Pojazd TO Pracownik;
GRANT SELECT ON `zakład_transportowy`.pracownik_pojazd TO Pracownik;
GRANT SELECT, UPDATE ON `zakład_transportowy`.Raport TO Pracownik;

GRANT SELECT ON `zakład_transportowy`.Pojazd TO Kierownik;
GRANT SELECT ON `zakład_transportowy`.Pracownik_Pojazd TO Kierownik;
GRANT SELECT, INSERT, UPDATE ON `zakład_transportowy`.Zamowienie TO Kierownik;
GRANT SELECT, INSERT, UPDATE ON `zakład_transportowy`.Kontakt TO Kierownik;
GRANT SELECT, INSERT, DELETE, UPDATE ON `zakład_transportowy`.Klient TO Kierownik;
GRANT SELECT, INSERT, UPDATE ON `zakład_transportowy`.Rozliczenie TO Kierownik;

GRANT SELECT ON `zakład_transportowy`.* TO Właściciel;
GRANT INSERT, UPDATE ON `zakład_transportowy`.Rozliczenie TO Właściciel;
GRANT INSERT, UPDATE, DELETE ON `zakład_transportowy`.Pracownik TO Właściciel;
GRANT INSERT, UPDATE, DELETE ON `zakład_transportowy`.Pojazd TO Właściciel;
GRANT INSERT, UPDATE, DELETE ON `zakład_transportowy`.Pracownik_pojazd TO Właściciel;
```

## 3.7 Wypełnianie tabel

```
-- -----------------------------------------------------
-- Data for table `zakład_transportowy`.`Pojazd`
-- -----------------------------------------------------

START TRANSACTION;
USE `zakład_transportowy`;
INSERT INTO `zakład_transportowy`.`Pojazd` (`idPojazd`, `NUMER_REJESTRACYJNY`, `RODZAJ`,
`DATA_OSTATNIEGO_PRZEGLADU`, `STAN_TECHNICZNY`) VALUES (1, 'DBL 45TA', 'BUS', '20.09.2020', 'SPRAWNY');
INSERT INTO `zakład_transportowy`.`Pojazd` (`idPojazd`, `NUMER_REJESTRACYJNY`, `RODZAJ`,
`DATA_OSTATNIEGO_PRZEGLADU`, `STAN_TECHNICZNY`) VALUES (2, 'DBL 12PO', 'BUS', '21.03.2017', 'POTRZEBNY
PRZEGLAD');
INSERT INTO `zakład_transportowy`.`Pojazd` (`idPojazd`, `NUMER_REJESTRACYJNY`, `RODZAJ`,
`DATA_OSTATNIEGO_PRZEGLADU`, `STAN_TECHNICZNY`) VALUES (3, 'DBL 12IO', 'CIĘŻARÓWKA', '31.12.2018', 'W
NAPRAWIE');
INSERT INTO `zakład_transportowy`.`Pojazd` (`idPojazd`, `NUMER_REJESTRACYJNY`, `RODZAJ`,
`DATA_OSTATNIEGO_PRZEGLADU`, `STAN_TECHNICZNY`) VALUES (4, 'DBL 99BD', 'CYSTERNA', '12.05.2020',
'SPRAWNY');
INSERT INTO `zakład_transportowy`.`Pojazd` (`idPojazd`, `NUMER_REJESTRACYJNY`, `RODZAJ`,
`DATA_OSTATNIEGO_PRZEGLADU`, `STAN_TECHNICZNY`) VALUES (5, 'DBL 98AM', 'OSOBÓWKA', '13.06.2017',
'POTRZEBNY PRZEGLAD');
INSERT INTO `zakład_transportowy`.`Pojazd` (`idPojazd`, `NUMER_REJESTRACYJNY`, `RODZAJ`,
`DATA_OSTATNIEGO_PRZEGLADU`, `STAN_TECHNICZNY`) VALUES (6, 'DBL 209A', 'TIR', '12.09.2020', 'SPRAWNY');
INSERT INTO `zakład_transportowy`.`Pojazd` (`idPojazd`, `NUMER_REJESTRACYJNY`, `RODZAJ`,
`DATA_OSTATNIEGO_PRZEGLADU`, `STAN_TECHNICZNY`) VALUES (7, 'DBL GL12', 'TIR', '10.10.2019', 'SPRAWNY');
INSERT INTO `zakład_transportowy`.`Pojazd` (`idPojazd`, `NUMER_REJESTRACYJNY`, `RODZAJ`,
`DATA_OSTATNIEGO_PRZEGLADU`, `STAN_TECHNICZNY`) VALUES (8, 'DBL 1092', 'TIR', '23.12.2019', 'DO
NAPRAWY');
INSERT INTO `zakład_transportowy`.`Pojazd` (`idPojazd`, `NUMER_REJESTRACYJNY`, `RODZAJ`,
`DATA_OSTATNIEGO_PRZEGLADU`, `STAN_TECHNICZNY`) VALUES (9, 'DBL 10AJ', 'OSOBÓWKA', '10.01.2020',
'SPRAWNY');
INSERT INTO `zakład_transportowy`.`Pojazd` (`idPojazd`, `NUMER_REJESTRACYJNY`, `RODZAJ`,
`DATA_OSTATNIEGO_PRZEGLADU`, `STAN_TECHNICZNY`) VALUES (10, 'DBL GAJ1', 'BUS', '12.08.2017', 'POTRZEBNY
PRZEGLAD');
COMMIT;
```

```
-- -----------------------------------------------------
-- Data for table `zakład_transportowy`.`Kontakt`
-- -----------------------------------------------------

START TRANSACTION;
USE `zakład_transportowy`;
INSERT INTO `zakład_transportowy`.`Kontakt` (`idKontakt`, `NUMER_TELEFONU`, `EMAIL`) VALUES (1,
'786108276', 'janusznosal@wp.pl');
INSERT INTO `zakład_transportowy`.`Kontakt` (`idKontakt`, `NUMER_TELEFONU`, `EMAIL`) VALUES (2,
'784391872', 'kamilnowak@wp.pl');
INSERT INTO `zakład_transportowy`.`Kontakt` (`idKontakt`, `NUMER_TELEFONU`, `EMAIL`) VALUES (3,
'609874192', 'aniajaka@gmail.com');
INSERT INTO `zakład_transportowy`.`Kontakt` (`idKontakt`, `NUMER_TELEFONU`, `EMAIL`) VALUES (4,
'781092876', 'edziamala@o2.pl');
INSERT INTO `zakład_transportowy`.`Kontakt` (`idKontakt`, `NUMER_TELEFONU`, `EMAIL`) VALUES (5,
'673109876', 'janekzielen@wp.pl');
INSERT INTO `zakład_transportowy`.`Kontakt` (`idKontakt`, `NUMER_TELEFONU`, `EMAIL`) VALUES (6,
'887209187', 'white2115@najlepszy.com');
INSERT INTO `zakład_transportowy`.`Kontakt` (`idKontakt`, `NUMER_TELEFONU`, `EMAIL`) VALUES (7,
'765109872', 'adispola@wp.pl');
INSERT INTO `zakład_transportowy`.`Kontakt` (`idKontakt`, `NUMER_TELEFONU`, `EMAIL`) VALUES (8,
'888109882', 'kubawit@op.pl');
INSERT INTO `zakład_transportowy`.`Kontakt` (`idKontakt`, `NUMER_TELEFONU`, `EMAIL`) VALUES (9,
'761998201', 'kornelekczlek@gmail.com');
INSERT INTO `zakład_transportowy`.`Kontakt` (`idKontakt`, `NUMER_TELEFONU`, `EMAIL`) VALUES (10,
'669109283', 'kamilwalczyk@gmail.com');
INSERT INTO `zakład_transportowy`.`Kontakt` (`idKontakt`, `NUMER_TELEFONU`, `EMAIL`) VALUES (11,
'605918230', 'lukaszburczy@gmail.com');
INSERT INTO `zakład_transportowy`.`Kontakt` (`idKontakt`, `NUMER_TELEFONU`, `EMAIL`) VALUES (12,
'600034128', 'jacusplacus@op.pl');
INSERT INTO `zakład_transportowy`.`Kontakt` (`idKontakt`, `NUMER_TELEFONU`, `EMAIL`) VALUES (14,
'765108287', 'damianrosja@gmail.com');
INSERT INTO `zakład_transportowy`.`Kontakt` (`idKontakt`, `NUMER_TELEFONU`, `EMAIL`) VALUES (13,
'569109280', 'adaskrol@wp.pl');
INSERT INTO `zakład_transportowy`.`Kontakt` (`idKontakt`, `NUMER_TELEFONU`, `EMAIL`) VALUES (15,
'604753918', 'agakrolik12@wp.pl');
INSERT INTO `zakład_transportowy`.`Kontakt` (`idKontakt`, `NUMER_TELEFONU`, `EMAIL`) VALUES (16,
'888666555', 'kowalkowalaa@wp.pl');
INSERT INTO `zakład_transportowy`.`Kontakt` (`idKontakt`, `NUMER_TELEFONU`, `EMAIL`) VALUES (17,
'607392189', NULL);
INSERT INTO `zakład_transportowy`.`Kontakt` (`idKontakt`, `NUMER_TELEFONU`, `EMAIL`) VALUES (18,
'698781979', 'zolikzolik@firmadobra.pl');
INSERT INTO `zakład_transportowy`.`Kontakt` (`idKontakt`, `NUMER_TELEFONU`, `EMAIL`) VALUES (19,
'989301837', 'wojciechmarsjanin@gmail.com');
INSERT INTO `zakład_transportowy`.`Kontakt` (`idKontakt`, `NUMER_TELEFONU`, `EMAIL`) VALUES (20,
'887909739', 'sebo@lysy.pl');
INSERT INTO `zakład_transportowy`.`Kontakt` (`idKontakt`, `NUMER_TELEFONU`, `EMAIL`) VALUES (21,
'888771083', 'szczyty@gory.com');
```

```
INSERT INTO `zakład_transportowy`.`Kontakt` (`idKontakt`, `NUMER_TELEFONU`, `EMAIL`) VALUES (22,
'666826398', 'beautyandthebeast@disney.pl');
INSERT INTO `zakład_transportowy`.`Kontakt` (`idKontakt`, `NUMER_TELEFONU`, `EMAIL`) VALUES (23,
'787979797', 'natallii@wp.pl');
INSERT INTO `zakład_transportowy`.`Kontakt` (`idKontakt`, `NUMER_TELEFONU`, `EMAIL`) VALUES (24,
'998808387', 'sowagrzes@las.pl');
INSERT INTO `zakład_transportowy`.`Kontakt` (`idKontakt`, `NUMER_TELEFONU`, `EMAIL`) VALUES (25,
'786766662', 'siedzenie@fotel.com');
INSERT INTO `zakład_transportowy`.`Kontakt` (`idKontakt`, `NUMER_TELEFONU`, `EMAIL`) VALUES (26,
'878787878', 'robertjanowski@metro.pl');
INSERT INTO `zakład_transportowy`.`Kontakt` (`idKontakt`, `NUMER_TELEFONU`, `EMAIL`) VALUES (27,
'998899889', 'kasiagroniec@metro.pl');
INSERT INTO `zakład_transportowy`.`Kontakt` (`idKontakt`, `NUMER_TELEFONU`, `EMAIL`) VALUES (28,
'717273774', 'pajac@cyrknakolkach.pl');
INSERT INTO `zakład_transportowy`.`Kontakt` (`idKontakt`, `NUMER_TELEFONU`, `EMAIL`) VALUES (29,
'887766554', 'placek@jacek.wpl');
INSERT INTO `zakład_transportowy`.`Kontakt` (`idKontakt`, `NUMER_TELEFONU`, `EMAIL`) VALUES (30,
'645327101', 'kasiabylamadra@plot.pl');
COMMIT;
```

```
-- -----------------------------------------------------
-- Data for table `zakład_transportowy`.`Klient`
-- -----------------------------------------------------

START TRANSACTION;
USE `zakład_transportowy`;
INSERT INTO `zakład_transportowy`.`Klient` (`idKlient`, `idKontakt`, `IMIE`, `NAZWISKO`, `LOGIN`,
`HASLO`, `OSTATNIE_LOGOWANIE`, `RODZAJ_KLIENTA`, `NAZWA_FIRMY`, `NIP`) VALUES
(1, 15, 'Anna', 'Król', 'aniakrolik', 'annneczka', '20.01.2020', 'OSOBA FIZYCZNA', NULL, NULL);
INSERT INTO `zakład_transportowy`.`Klient` (`idKlient`, `idKontakt`, `IMIE`, `NAZWISKO`, `LOGIN`,
`HASLO`, `OSTATNIE_LOGOWANIE`, `RODZAJ_KLIENTA`, `NAZWA_FIRMY`, `NIP`) VALUES
(2, 16, 'Agata', 'Kowal', 'agaciak', 'agiskowalik', '10.05.2020', 'OSOBA FIZYCZNA', NULL, NULL);
INSERT INTO `zakład_transportowy`.`Klient` (`idKlient`, `idKontakt`, `IMIE`, `NAZWISKO`, `LOGIN`,
`HASLO`, `OSTATNIE_LOGOWANIE`, `RODZAJ_KLIENTA`, `NAZWA_FIRMY`, `NIP`) VALUES
(3, 17, 'Karolina', 'Skowronek', 'karaq0', 'kar019@', '14.01.2019', 'OSOBA FIZYCZNA', NULL, NULL);
INSERT INTO `zakład_transportowy`.`Klient` (`idKlient`, `idKontakt`, `IMIE`, `NAZWISKO`, `LOGIN`,
`HASLO`, `OSTATNIE_LOGOWANIE`, `RODZAJ_KLIENTA`, `NAZWA_FIRMY`, `NIP`) VALUES
(4, 18, 'Partryk', 'Zolik', 'papis', 'patrykzol12', '13.01.2020', 'FIRMA', 'ZOZOL2109', '9871092871');
INSERT INTO `zakład_transportowy`.`Klient` (`idKlient`, `idKontakt`, `IMIE`, `NAZWISKO`, `LOGIN`,
`HASLO`, `OSTATNIE_LOGOWANIE`, `RODZAJ_KLIENTA`, `NAZWA_FIRMY`, `NIP`) VALUES
(5, 19, 'Wojtek', 'Mars', 'maRSIO', 'rsiomark', '12.11.2020', 'OSOBA FIZYCZNA', NULL, NULL);
INSERT INTO `zakład_transportowy`.`Klient` (`idKlient`, `idKontakt`, `IMIE`, `NAZWISKO`, `LOGIN`,
`HASLO`, `OSTATNIE_LOGOWANIE`, `RODZAJ_KLIENTA`, `NAZWA_FIRMY`, `NIP`) VALUES
 (6, 20, 'Sebastian', 'Wasiak', 'wasioaseba', 'seba34', '09.09.2020', 'OSOBA FIZYCZNA', NULL, NULL);
INSERT INTO `zakład_transportowy`.`Klient` (`idKlient`, `idKontakt`, `IMIE`, `NAZWISKO`, `LOGIN`,
`HASLO`, `OSTATNIE_LOGOWANIE`, `RODZAJ_KLIENTA`, `NAZWA_FIRMY`, `NIP`) VALUES
(7, 21, 'Radosław', 'Góra', 'gorekradek', 'szczyty', '12.03.2019', 'OSOBA FIZYCZNA', NULL, NULL);
INSERT INTO `zakład_transportowy`.`Klient` (`idKlient`, `idKontakt`, `IMIE`, `NAZWISKO`, `LOGIN`,
`HASLO`, `OSTATNIE_LOGOWANIE`, `RODZAJ_KLIENTA`, `NAZWA_FIRMY`, `NIP`) VALUES
(8, 22, 'Filip', 'Bestia', 'beastfifi', 'fifrk123', '22.06.2020', 'OSOBA FIZYCZNA', NULL, NULL);
INSERT INTO `zakład_transportowy`.`Klient` (`idKlient`, `idKontakt`, `IMIE`, `NAZWISKO`, `LOGIN`,
`HASLO`, `OSTATNIE_LOGOWANIE`, `RODZAJ_KLIENTA`, `NAZWA_FIRMY`, `NIP`) VALUES (9, 23, 'Natalia',
'Karawanik', 'karolik', 'kora451', '12.12.2019', 'FIRMA', 'Koraliki Natalci', '1982001982');
INSERT INTO `zakład_transportowy`.`Klient` (`idKlient`, `idKontakt`, `IMIE`, `NAZWISKO`, `LOGIN`,
`HASLO`, `OSTATNIE_LOGOWANIE`, `RODZAJ_KLIENTA`, `NAZWA_FIRMY`, `NIP`) VALUES (10, 24, 'Grzegorz',
'Sówka', 'skkawegrz', 'grzesiuskk', '02.02.2018', 'FIRMA', 'Sowa 09', '1028792910');
INSERT INTO `zakład_transportowy`.`Klient` (`idKlient`, `idKontakt`, `IMIE`, `NAZWISKO`, `LOGIN`,
`HASLO`, `OSTATNIE_LOGOWANIE`, `RODZAJ_KLIENTA`, `NAZWA_FIRMY`, `NIP`) VALUES (11, 25, 'Marian', 'Fotel',
'krzeslomarys', 'mariansiedzenie', '06.04.2020', 'OSOBA FIZYCZNA', NULL, NULL);
INSERT INTO `zakład_transportowy`.`Klient` (`idKlient`, `idKontakt`, `IMIE`, `NAZWISKO`, `LOGIN`,
`HASLO`, `OSTATNIE_LOGOWANIE`, `RODZAJ_KLIENTA`, `NAZWA_FIRMY`, `NIP`) VALUES (12, 26, 'Robert',
'Janowski', 'robcio', 'metro', '09.08.2019', 'FIRMA ', 'A JA NIE, PO PROSTU NIE ', '1973901903');
INSERT INTO `zakład_transportowy`.`Klient` (`idKlient`, `idKontakt`, `IMIE`, `NAZWISKO`, `LOGIN`,
`HASLO`, `OSTATNIE_LOGOWANIE`, `RODZAJ_KLIENTA`, `NAZWA_FIRMY`, `NIP`) VALUES (13, 27, 'Katarzyna ',
'Groniec', 'kaskabaska', 'metro42', '08.12.2019', 'FIRMA', 'TAFLA SZKŁA', '1028662891');
INSERT INTO `zakład_transportowy`.`Klient` (`idKlient`, `idKontakt`, `IMIE`, `NAZWISKO`, `LOGIN`,
`HASLO`, `OSTATNIE_LOGOWANIE`, `RODZAJ_KLIENTA`, `NAZWA_FIRMY`, `NIP`) VALUES (14, 28, 'Daniel', 'Pajac',
'cyrkicyki', 'cykadyDAniela', '14.12.2018', 'FIRMA', 'CYRK NA KÓŁKACH', '0381719374');
```

```
INSERT INTO `zakład_transportowy`.`Klient` (`idKlient`, `idKontakt`, `IMIE`, `NAZWISKO`, `LOGIN`,
`HASLO`, `OSTATNIE_LOGOWANIE`, `RODZAJ_KLIENTA`, `NAZWA_FIRMY`, `NIP`) VALUES (15, 29, 'Jacek', 'Placek',
'naOlejuMamaKrzyczy', 'StojZlodzieju', '28.02.2019', 'FIRMA', 'TłUSZCZ I SMALEC', '9462810181');
INSERT INTO `zakład_transportowy`.`Klient` (`idKlient`, `idKontakt`, `IMIE`, `NAZWISKO`, `LOGIN`,
`HASLO`, `OSTATNIE_LOGOWANIE`, `RODZAJ_KLIENTA`, `NAZWA_FIRMY`, `NIP`) VALUES (16, 30, 'Katarzyna',
'Mądra', 'bylaidziurewplocie', 'wywiercila', '13.09.2019', 'OSOBA FIZYCZNA', NULL, NULL);
COMMIT;



-- -----------------------------------------------------
-- Data for table `zakład_transportowy`.`Zamowienie`
-- -----------------------------------------------------

START TRANSACTION;
USE `zakład_transportowy`;
INSERT INTO `zakład_transportowy`.`Zamowienie` (`idZamowienie`, `idKlient`, `idPojazd`, `STATUS`,
`DATA_ZLOZENIA`, `DATA_REALIZACJI`, `CENA`, `CZY_ZAPLACONO`) VALUES
(1, 4, 3, 'ZREALIZOWANO', '20.09.10', '20.10.11', 109.29, 'TAK');
INSERT INTO `zakład_transportowy`.`Zamowienie` (`idZamowienie`, `idKlient`, `idPojazd`, `STATUS`,
`DATA_ZLOZENIA`, `DATA_REALIZACJI`, `CENA`, `CZY_ZAPLACONO`) VALUES
(2, 4, 9, 'W TRAKCIE', '20.09.20', '20.11.18', 210.09, 'NIE');
INSERT INTO `zakład_transportowy`.`Zamowienie` (`idZamowienie`, `idKlient`, `idPojazd`, `STATUS`,
`DATA_ZLOZENIA`, `DATA_REALIZACJI`, `CENA`, `CZY_ZAPLACONO`) VALUES
 (3, 2, 1, 'ZREALIZOWANO', '19.08.09', '20.01.14', 300.10, 'TAK');
INSERT INTO `zakład_transportowy`.`Zamowienie` (`idZamowienie`, `idKlient`, `idPojazd`, `STATUS`,
`DATA_ZLOZENIA`, `DATA_REALIZACJI`, `CENA`, `CZY_ZAPLACONO`) VALUES
(4, 15, 4, 'ZLOZONO', '20.06.25', '20.12.21', 100.59, 'NIE');
INSERT INTO `zakład_transportowy`.`Zamowienie` (`idZamowienie`, `idKlient`, `idPojazd`, `STATUS`,
`DATA_ZLOZENIA`, `DATA_REALIZACJI`, `CENA`, `CZY_ZAPLACONO`) VALUES
(5, 13, 10, 'ZREALIZOWANO', '15.06.19', '15.07.23', 500.89, 'TAK');
INSERT INTO `zakład_transportowy`.`Zamowienie` (`idZamowienie`, `idKlient`, `idPojazd`, `STATUS`,
`DATA_ZLOZENIA`, `DATA_REALIZACJI`, `CENA`, `CZY_ZAPLACONO`) VALUES
 (6, 1, 4, 'ZREALIZOWANO', '20.09.15', '20.10.05', 897.10, 'TAK');
INSERT INTO `zakład_transportowy`.`Zamowienie` (`idZamowienie`, `idKlient`, `idPojazd`, `STATUS`,
`DATA_ZLOZENIA`, `DATA_REALIZACJI`, `CENA`, `CZY_ZAPLACONO`) VALUES
(7, 10, 6, 'W TRAKCIE', '20.10.01', '20.11.19', 1000.89, 'NIE');
COMMIT;
```

```
-- -----------------------------------------------------
-- Data for table `zakład_transportowy`.`Pracownik`
-- -----------------------------------------------------
START TRANSACTION;
USE `zakład_transportowy`;
INSERT INTO `zakład_transportowy`.`Pracownik` (`idPracownik`, `idKontakt`, `idKierownik`, `IMIE`,
`NAZWISKO`, `LOGIN`, `HASLO`, `UPRAWNIENIA`, `PENSJA`, `DATA_ZATRUDNIENIA`) VALUES (1, 1, NULL, 'Janusz',
'Nosal', 'nosaljanek23', 'JANNOS3', 'WLASCICIEL', 10000, '01.01.2009');
INSERT INTO `zakład_transportowy`.`Pracownik` (`idPracownik`, `idKontakt`, `idKierownik`, `IMIE`,
`NAZWISKO`, `LOGIN`, `HASLO`, `UPRAWNIENIA`, `PENSJA`, `DATA_ZATRUDNIENIA`) VALUES (2, 2, 2, 'Kamil',
'Nowak', 'komilnow', 'Joma23', 'MECHANIK', 3000, '03.05.2010');
INSERT INTO `zakład_transportowy`.`Pracownik` (`idPracownik`, `idKontakt`, `idKierownik`, `IMIE`,
`NAZWISKO`, `LOGIN`, `HASLO`, `UPRAWNIENIA`, `PENSJA`, `DATA_ZATRUDNIENIA`) VALUES (3, 3, 1, 'Anna',
'Jakaś', 'jakaanna', 'annajaka', 'KIEROWNIK', 5000, '13.04.2011');
INSERT INTO `zakład_transportowy`.`Pracownik` (`idPracownik`, `idKontakt`, `idKierownik`, `IMIE`,
`NAZWISKO`, `LOGIN`, `HASLO`, `UPRAWNIENIA`, `PENSJA`, `DATA_ZATRUDNIENIA`) VALUES (4, 4, 2, 'Edyta',
'Wilk', 'wilkaaaD', 'EWilka', 'MECHANIK', 4000, '14.05.2018');
INSERT INTO `zakład_transportowy`.`Pracownik` (`idPracownik`, `idKontakt`, `idKierownik`, `IMIE`,
`NAZWISKO`, `LOGIN`, `HASLO`, `UPRAWNIENIA`, `PENSJA`, `DATA_ZATRUDNIENIA`) VALUES (5, 5, 2, 'Jan',
'Zielony', 'zielonek32', 'zielenmoimkolorem', 'KIEROWCA', 3500, '03.02.2017');
INSERT INTO `zakład_transportowy`.`Pracownik` (`idPracownik`, `idKontakt`, `idKierownik`, `IMIE`,
`NAZWISKO`, `LOGIN`, `HASLO`, `UPRAWNIENIA`, `PENSJA`, `DATA_ZATRUDNIENIA`) VALUES (6, 6, 2, 'Tomasz',
'Białas', 'lyseczolo10101', 'bialaglowa23', 'KIEROWCA', 3600, '01.01.2013');
INSERT INTO `zakład_transportowy`.`Pracownik` (`idPracownik`, `idKontakt`, `idKierownik`, `IMIE`,
`NAZWISKO`, `LOGIN`, `HASLO`, `UPRAWNIENIA`, `PENSJA`, `DATA_ZATRUDNIENIA`) VALUES (7, 7, 2, 'Adrian',
'Opał', 'adisopala', 'opal56', 'KIEROWCA', 2000, '03.11.2020');
INSERT INTO `zakład_transportowy`.`Pracownik` (`idPracownik`, `idKontakt`, `idKierownik`, `IMIE`,
`NAZWISKO`, `LOGIN`, `HASLO`, `UPRAWNIENIA`, `PENSJA`, `DATA_ZATRUDNIENIA`) VALUES (8, 8, 2, 'Jakub',
'Witkos', 'witkacy10', 'witkos20', 'KIEROWCA', 4100, '01.03.2009');
INSERT INTO `zakład_transportowy`.`Pracownik` (`idPracownik`, `idKontakt`, `idKierownik`, `IMIE`,
`NAZWISKO`, `LOGIN`, `HASLO`, `UPRAWNIENIA`, `PENSJA`, `DATA_ZATRUDNIENIA`) VALUES (9, 9, 2, 'Kornel',
'Człowiek', 'humanre', 'aimhman', 'KIEROWCA', 3600, '14.01.2005');
INSERT INTO `zakład_transportowy`.`Pracownik` (`idPracownik`, `idKontakt`, `idKierownik`, `IMIE`,
`NAZWISKO`, `LOGIN`, `HASLO`, `UPRAWNIENIA`, `PENSJA`, `DATA_ZATRUDNIENIA`) VALUES (10, 10, 2, 'Kamil',
'Walak', 'walekaa', 'kamil222', 'KIEROWCA', 2800, '13.12.2017');
INSERT INTO `zakład_transportowy`.`Pracownik` (`idPracownik`, `idKontakt`, `idKierownik`, `IMIE`,
`NAZWISKO`, `LOGIN`, `HASLO`, `UPRAWNIENIA`, `PENSJA`, `DATA_ZATRUDNIENIA`) VALUES (11, 11, 2, 'Łukasz',
'Burczyk', 'bbuuela', 'burdyn@30', 'KIEROWCA', 3000, '15.10.2018');
INSERT INTO `zakład_transportowy`.`Pracownik` (`idPracownik`, `idKontakt`, `idKierownik`, `IMIE`,
`NAZWISKO`, `LOGIN`, `HASLO`, `UPRAWNIENIA`, `PENSJA`, `DATA_ZATRUDNIENIA`) VALUES (12, 12, 2, 'Jacek',
'Ucho', 'uszypszczola', 'nosuchopieta', 'KIEROWCA', 4200, '01.03.2010');
INSERT INTO `zakład_transportowy`.`Pracownik` (`idPracownik`, `idKontakt`, `idKierownik`, `IMIE`,
`NAZWISKO`, `LOGIN`, `HASLO`, `UPRAWNIENIA`, `PENSJA`, `DATA_ZATRUDNIENIA`) VALUES (13, 13, 2, 'Adam',
'Wpisany', 'wpisanyzostal', 'wpisalemsiesam', 'KIEROWCA', 3200, '04.04.2014');
INSERT INTO `zakład_transportowy`.`Pracownik` (`idPracownik`, `idKontakt`, `idKierownik`, `IMIE`,
`NAZWISKO`, `LOGIN`, `HASLO`, `UPRAWNIENIA`, `PENSJA`, `DATA_ZATRUDNIENIA`) VALUES (14, 14, 2, 'Damian',
'Rusek', 'ruskiczlowiek', 'nastepnyde', 'KIEROWCA', 5000, '01.01.2011');
COMMIT;
```

```
-- -----------------------------------------------------
-- Data for table `zakład_transportowy`.`Rozliczenie`
-- -----------------------------------------------------

START TRANSACTION;
USE `zakład_transportowy`;
INSERT INTO `zakład_transportowy`.`Rozliczenie` (`idRozliczenie`, `idZamowienie`, `FORMA_ROZLICZENIA`,
`FORMA_PLATNOSCI`, `DATA_ROZLICZENIA`, `BRUTTO`, `NETTO`, `VAT`) VALUES (1, 1, 'PARAGON', 'GOTOWKA',
(SELECT `DATA_REALIZACJI` FROM `Zamowienie` WHERE `idZamowienie` = 1), (SELECT `CENA` FROM `Zamowienie`
WHERE `idZamowienie` = 1), netto(`BRUTTO`), vat(`BRUTTO`));
INSERT INTO `zakład_transportowy`.`Rozliczenie` (`idRozliczenie`, `idZamowienie`, `FORMA_ROZLICZENIA`,
`FORMA_PLATNOSCI`, `DATA_ROZLICZENIA`, `BRUTTO`, `NETTO`, `VAT`) VALUES (2, 3, 'PARAGON', 'KARTA
PLATNICZA', (SELECT `DATA_REALIZACJI` FROM `Zamowienie` WHERE `idZamowienie` = 3), (SELECT `CENA` FROM
`Zamowienie` WHERE `idZamowienie` = 3), netto(`BRUTTO`), vat(`BRUTTO`));
INSERT INTO `zakład_transportowy`.`Rozliczenie` (`idRozliczenie`, `idZamowienie`, `FORMA_ROZLICZENIA`,
`FORMA_PLATNOSCI`, `DATA_ROZLICZENIA`, `BRUTTO`, `NETTO`, `VAT`) VALUES (3, 5, 'FAKTURA', 'PRZELEW',
(SELECT `DATA_REALIZACJI` FROM `Zamowienie` WHERE `idZamowienie` = 5), (SELECT `CENA` FROM `Zamowienie`
WHERE `idZamowienie` = 5), netto(`BRUTTO`), vat(`BRUTTO`));
INSERT INTO `zakład_transportowy`.`Rozliczenie` (`idRozliczenie`, `idZamowienie`, `FORMA_ROZLICZENIA`,
`FORMA_PLATNOSCI`, `DATA_ROZLICZENIA`, `BRUTTO`, `NETTO`, `VAT`) VALUES (4, 6, 'PARAGON', 'GOTOWKA',
(SELECT `DATA_REALIZACJI` FROM `Zamowienie` WHERE `idZamowienie` = 6), (SELECT `CENA` FROM `Zamowienie`
WHERE `idZamowienie` = 6), netto(`BRUTTO`), vat(`BRUTTO`));
COMMIT;



-- -----------------------------------------------------
-- Data for table `zakład_transportowy`.`Raport`
-- -----------------------------------------------------
START TRANSACTION;
USE `zakład_transportowy`;
INSERT INTO `zakład_transportowy`.`Raport` (`idRaport`, `idPracownik`, `idZamowienie`) VALUES
(1, 2, 1);
INSERT INTO `zakład_transportowy`.`Raport` (`idRaport`, `idPracownik`, `idZamowienie`) VALUES
(2, 3, 2);
INSERT INTO `zakład_transportowy`.`Raport` (`idRaport`, `idPracownik`, `idZamowienie`) VALUES
(3, 4, 3);
INSERT INTO `zakład_transportowy`.`Raport` (`idRaport`, `idPracownik`, `idZamowienie`) VALUES
(4, 5, 4);
INSERT INTO `zakład_transportowy`.`Raport` (`idRaport`, `idPracownik`, `idZamowienie`) VALUES
(5, 6, 5);
INSERT INTO `zakład_transportowy`.`Raport` (`idRaport`, `idPracownik`, `idZamowienie`) VALUES
 (6, 7, 6);
INSERT INTO `zakład_transportowy`.`Raport` (`idRaport`, `idPracownik`, `idZamowienie`) VALUES
 (7, 8, 7);
COMMIT;
```

```sql
-- -----------------------------------------------------
-- Data for table `zakład_transportowy`.`Pracownik_Pojazd`
-- -----------------------------------------------------
START TRANSACTION;
USE `zakład_transportowy`;
INSERT INTO `zakład_transportowy`.`Pracownik_Pojazd` (`idPracownik_Pojazd`, `idPojazd`, `idPracownik`)
VALUES (1, 4, 5);
INSERT INTO `zakład_transportowy`.`Pracownik_Pojazd` (`idPracownik_Pojazd`, `idPojazd`, `idPracownik`)
VALUES (2, 6, 5);
INSERT INTO `zakład_transportowy`.`Pracownik_Pojazd` (`idPracownik_Pojazd`, `idPojazd`, `idPracownik`)
VALUES (3, 7, 5);
INSERT INTO `zakład_transportowy`.`Pracownik_Pojazd` (`idPracownik_Pojazd`, `idPojazd`, `idPracownik`)
VALUES (4, 8, 5);
INSERT INTO `zakład_transportowy`.`Pracownik_Pojazd` (`idPracownik_Pojazd`, `idPojazd`, `idPracownik`)
VALUES (5, 4, 6);
INSERT INTO `zakład_transportowy`.`Pracownik_Pojazd` (`idPracownik_Pojazd`, `idPojazd`, `idPracownik`)
VALUES (6, 6, 6);
INSERT INTO `zakład_transportowy`.`Pracownik_Pojazd` (`idPracownik_Pojazd`, `idPojazd`, `idPracownik`)
VALUES (7, 7, 6);
INSERT INTO `zakład_transportowy`.`Pracownik_Pojazd` (`idPracownik_Pojazd`, `idPojazd`, `idPracownik`)
VALUES (8, 8, 6);
INSERT INTO `zakład_transportowy`.`Pracownik_Pojazd` (`idPracownik_Pojazd`, `idPojazd`, `idPracownik`)
VALUES (9, 1, 7);
INSERT INTO `zakład_transportowy`.`Pracownik_Pojazd` (`idPracownik_Pojazd`, `idPojazd`, `idPracownik`)
VALUES (10, 2, 7);
INSERT INTO `zakład_transportowy`.`Pracownik_Pojazd` (`idPracownik_Pojazd`, `idPojazd`, `idPracownik`)
VALUES (11, 10, 7);
INSERT INTO `zakład_transportowy`.`Pracownik_Pojazd` (`idPracownik_Pojazd`, `idPojazd`, `idPracownik`)
VALUES (12, 3, 8);
INSERT INTO `zakład_transportowy`.`Pracownik_Pojazd` (`idPracownik_Pojazd`, `idPojazd`, `idPracownik`)
VALUES (13, 5, 8);
INSERT INTO `zakład_transportowy`.`Pracownik_Pojazd` (`idPracownik_Pojazd`, `idPojazd`, `idPracownik`)
VALUES (14, 9, 8);
INSERT INTO `zakład_transportowy`.`Pracownik_Pojazd` (`idPracownik_Pojazd`, `idPojazd`, `idPracownik`)
VALUES (15, 3, 10);
INSERT INTO `zakład_transportowy`.`Pracownik_Pojazd` (`idPracownik_Pojazd`, `idPojazd`, `idPracownik`)
VALUES (16, 1, 10);
INSERT INTO `zakład_transportowy`.`Pracownik_Pojazd` (`idPracownik_Pojazd`, `idPojazd`, `idPracownik`)
VALUES (17, 2, 10);
INSERT INTO `zakład_transportowy`.`Pracownik_Pojazd` (`idPracownik_Pojazd`, `idPojazd`, `idPracownik`)
VALUES (18, 3, 11);
INSERT INTO `zakład_transportowy`.`Pracownik_Pojazd` (`idPracownik_Pojazd`, `idPojazd`, `idPracownik`)
VALUES (19, 4, 11);
INSERT INTO `zakład_transportowy`.`Pracownik_Pojazd` (`idPracownik_Pojazd`, `idPojazd`, `idPracownik`)
VALUES (20, 1, 12);
INSERT INTO `zakład_transportowy`.`Pracownik_Pojazd` (`idPracownik_Pojazd`, `idPojazd`, `idPracownik`)
VALUES (21, 2, 12);
```

```
INSERT INTO `zakład_transportowy`.`Pracownik_Pojazd` (`idPracownik_Pojazd`, `idPojazd`, `idPracownik`)
VALUES (22, 10, 12);
INSERT INTO `zakład_transportowy`.`Pracownik_Pojazd` (`idPracownik_Pojazd`, `idPojazd`, `idPracownik`)
VALUES (23, 5, 12);
INSERT INTO `zakład_transportowy`.`Pracownik_Pojazd` (`idPracownik_Pojazd`, `idPojazd`, `idPracownik`)
VALUES (24, 9, 12);
INSERT INTO `zakład_transportowy`.`Pracownik_Pojazd` (`idPracownik_Pojazd`, `idPojazd`, `idPracownik`)
VALUES (25, 6, 13);
INSERT INTO `zakład_transportowy`.`Pracownik_Pojazd` (`idPracownik_Pojazd`, `idPojazd`, `idPracownik`)
VALUES (26, 3, 14);
COMMIT;
```

## 3.8 Zapytania CRUD

```
-- -----------------------------------------------------
Zapytania CRUD dla tabeli Pojazd
-- -----------------------------------------------------

INSERT INTO `Pojazd` (`NUMER_REJESTRACYJNY`, `RODZAJ`, `DATA_OSTATNIEGO_PRZEGLADU`, `STAN_TECHNICZNY`)
VALUES ('WR 58PA', 'CYSTERNA', '20.09.10', 'SPRAWNY');
SELECT * FROM `pojazd`;
UPDATE `pojazd` SET `STAN_TECHNICZNY` = 'DO NAPRAWY' WHERE `idPojazd`  = 7;
DELETE FROM `pojazd` WHERE `idPojazd`   = 11;
```

Wyniki zapytań przedstawiono w tabeli:

| Zapytanie | Wynik |
|---|---|
| **INSERT** | 1 row(s) affected |
| **SELECT** | 11 rows(s) returned |
| **UPDATE** | 1 row(s) affected Rows matched: 1 Changed 1 Warnings 0 |
| **DELETE** | 1 row(s) affected |

```
-- -----------------------------------------------------
Zapytania CRUD dla tabeli Kontakt
-- -----------------------------------------------------

INSERT INTO `Kontakt` (`NUMER_TELEFONU`, `EMAIL`) VALUES ('523409276', 'tomaszKruk@gmail.com');
SELECT `EMAIL` FROM `kontakt`;
UPDATE `Kontakt` SET `NUMER_TELEFONU` = '524097276' WHERE `NUMER_TELEFONU` = '523409276';
DELETE FROM `kontakt` WHERE `idKontakt` = 31;
```

Wyniki zapytań przedstawiono w tabeli:

| Zapytanie | Wynik |
|---|---|
| **INSERT** | 1 row(s) affected |
| **SELECT** | 31 rows(s) returned |
| **UPDATE** | 1 row(s) affected Rows matched: 1 Changed 1 Warnings 0 |
| **DELETE** | 1 row(s) affected |

```
-- -----------------------------------------------------
Zapytania CRUD dla tabeli Klient
-- -----------------------------------------------------

INSERT INTO `Kontakt` (`idKontakt`, `NUMER_TELEFONU`, `EMAIL`) VALUES (31, '523409276',
'tomaszKruk@gmail.com');
INSERT INTO `zakład_transportowy`.`Klient` (`idKontakt`, `IMIE`, `NAZWISKO`, `LOGIN`, `HASLO`,
`OSTATNIE_LOGOWANIE`, `RODZAJ_KLIENTA`, `NAZWA_FIRMY`, `NIP`) VALUES (31, 'Monika', 'Mop', 'mops12',
'monia893', '20.10.18', 'OSOBA FIZYCZNA', NULL, NULL);
SELECT * FROM `klient` WHERE `RODZAJ_KLIENTA` = 'OSOBA FIZYCZNA';
UPDATE `klient` SET `HASLO` = 'monika12' WHERE `idKlient` = 17;
DELETE FROM `klient` WHERE `idKlient`  = 17;
```

Wyniki zapytań przedstawiono w tabeli:

| Zapytanie | Wynik |
|---|---|
| **INSERT** | 1 row(s) affected<br>1 row(s) affected |
| **SELECT** | 10 rows(s) returned |
| **UPDATE** | 1 row(s) affected Rows matched: 1 Changed 1 Warnings 0 |
| **DELETE** | 1 row(s) affected |

```
-- -----------------------------------------------------
Zapytania CRUD dla tabeli Pracownik
-- -----------------------------------------------------

INSERT INTO `Kontakt` (`idKontakt`, `NUMER_TELEFONU`, `EMAIL`) VALUES (31, '523409276',
'tomaszKruk@gmail.com');
INSERT INTO `Pracownik` (`idPracownik`, `idKontakt`, `idKierownik`, `IMIE`, `NAZWISKO`, `LOGIN`, `HASLO`,
`UPRAWNIENIA`, `PENSJA`, `DATA_ZATRUDNIENIA`) VALUES (15, 31, 2, 'Tomasz', 'Kruk', 'tKruk', 'Ptak111',
'KIEROWCA', 4000, '20.11.21');
SELECT * FROM `Pracownik`;
UPDATE `Pracownik` SET `PENSJA` = `PENSJA` + 300 WHERE `UPRAWNIENIA` = 'KIEROWCA';
DELETE FROM `pracownik` WHERE `idPracownik` = 15;
```

Wyniki zapytań przedstawiono w tabeli:

| Zapytanie | Wynik |
|---|---|
| **INSERT** | 1 row(s) affected<br>1 row(s) affected |
| **SELECT** | 15 rows(s) returned |
| **UPDATE** | 11 row(s) affected Rows matched: 11 Changed 11 Warnings 0 |
| **DELETE** | 1 row(s) affected |

```
-- -----------------------------------------------------
Zapytania CRUD dla tabeli Zamowienie
-- -----------------------------------------------------

INSERT INTO `Zamowienie` (`idZamowienie`, `idKlient`, `idPojazd`, `STATUS`, `DATA_ZLOZENIA`,
`DATA_REALIZACJI`, `CENA`, `CZY_ZAPLACONO`) VALUES (8, 6, 10, 'ZLOZONO', '20.11.21', '20.01.05', 356.78,
'NIE');
SELECT * FROM `Zamowienie`;
UPDATE `Zamowienie` SET `CZY_ZAPLACONO` = 'TAK';
DELETE FROM `Zamowienie` WHERE `idZamowienie` = 8;
```

Wyniki zapytań przedstawiono w tabeli:

| Zapytanie | Wynik |
|---|---|
| **INSERT** | 1 row(s) affected |
| **SELECT** | 8 rows(s) returned |
| **UPDATE** | 4 row(s) affected Rows matched: 4 Changed 4 Warnings 0 |
| **DELETE** | 1 row(s) affected |

```
-- -----------------------------------------------------
Zapytania CRUD dla tabeli Rozliczenie
-- -----------------------------------------------------

INSERT INTO `Rozliczenie` (`idRozliczenie`, `idZamowienie`, `FORMA_ROZLICZENIA`, `FORMA_PLATNOSCI`,
`DATA_ROZLICZENIA`, `BRUTTO`, `NETTO`, `VAT`) VALUES (5, 7, 'PARAGON', 'GOTOWKA', (SELECT
`DATA_REALIZACJI` FROM `Zamowienie` WHERE `idZamowienie` = 7), (SELECT `CENA` FROM `Zamowienie` WHERE
`idZamowienie` = 7), netto(`BRUTTO`),vat(`BRUTTO`));
SELECT * FROM `Rozliczenie` WHERE `FORMA_PLATNOSCI` = 'PRZELEW';
UPDATE `Rozliczenie` SET `FORMA_ROZLICZENIA` = 'FAKTURA' WHERE `idRozliczenie` = 1;
DELETE FROM `Rozliczenie` WHERE `idRozliczenie` = 4;
```

Wyniki zapytań przedstawiono w tabeli:

| Zapytanie | Wynik |
|---|---|
| **INSERT** | 1 row(s) affected |
| **SELECT** | 1 rows(s) returned |
| **UPDATE** | 1 row(s) affected Rows matched: 1 Changed 1 Warnings 0 |
| **DELETE** | 1 row(s) affected |

```sql
-- -----------------------------------------------------
Zapytania CRUD dla tabeli Pracownik_Pojazd
-- -----------------------------------------------------

INSERT INTO `Pracownik_Pojazd` (`idPracownik_Pojazd`, `idPojazd`, `idPracownik`) VALUES (27, 8, 8);
SELECT * FROM `Pracownik_Pojazd`;
UPDATE `Pracownik_Pojazd` SET `idPojazd` = 7 WHERE `idPracownik` = 3;
DELETE FROM `Pracownik_Pojazd` WHERE `idPracownik` > 10;
```

Wyniki zapytań przedstawiono w tabeli:

| Zapytanie | Wynik |
|---|---|
| **INSERT** | 1 row(s) affected |
| **SELECT** | 27 rows(s) returned |
| **UPDATE** | 3 row(s) affected Rows matched: 4 Changed 3 Warnings 0 |
| **DELETE** | 9 row(s) affected |

```sql
-- -----------------------------------------------------
Zapytania CRUD dla tabeli Raport
-- -----------------------------------------------------

INSERT INTO `Raport` (`idRaport`, `idPracownik`, `idZamowienie`) VALUES (8, 2, 3);
SELECT * FROM `Raport`;
UPDATE `Raport` SET `idPracownik` = 7 WHERE `idZamowienie` = 4;
DELETE FROM `Raport` WHERE `idPracownik` < 4;
```

Wyniki zapytań przedstawiono w tabeli:

| Zapytanie | Wynik |
|---|---|
| **INSERT** | 1 row(s) affected |
| **SELECT** | 8 rows(s) returned |
| **UPDATE** | 1 row(s) affected Rows matched: 1 Changed 1 Warnings 0 |
| **DELETE** | 3 row(s) affected |

## 3.9 Test wyzwalaczy

```
-- -----------------------------------------------------
Test wyzwalacza usuniecie_konta_klienta
-- -----------------------------------------------------

SELECT * FROM `Kontakt` WHERE `idKontakt` = (SELECT `idKontakt` FROM `Klient` WHERE `idKlient` = 3);
DELETE FROM `Klient` WHERE `idKlient` = 3;
SELECT * FROM `Klient` WHERE `idKlient` = 3;
SELECT * FROM `Zamowienie` WHERE `idKlient` = 3;
SELECT * FROM `Kontakt` WHERE `idKontakt` = 16;



-- -----------------------------------------------------
Test wyzwalacza usuniecie_konta_pracownika
-- -----------------------------------------------------

SELECT * FROM `Kontakt` WHERE `idKontakt` = (SELECT `idKontakt` FROM `Pracownik`
WHERE `idPracownik` = 7);
DELETE FROM `Pracownik` WHERE `idPracownik` = 7;
SELECT * FROM `Pracownik` WHERE `idPracownik` = 7;
SELECT * FROM `Pracownik_Pojazd` WHERE `idPracownik` = 7;
SELECT * FROM `Raport` WHERE `idPracownik` = 7;
SELECT * FROM `Kontakt` WHERE `idKontakt` = 7;



-- -----------------------------------------------------
Test wyzwalacza usuniecie_pojazdu
-- -----------------------------------------------------

DELETE FROM `Pojazd` WHERE `idPojazd` = 3;
SELECT * FROM `Pojazd` WHERE `idPojazd` = 3;
SELECT * FROM `Pracownik_Pojazd` WHERE `idPojazd` = 3;
```

**Wszystkie wyzwalacze działają zgodnie z założeniami**

## 3.10 Polityka bezpieczeństwa

| Użytkownik | Zapytanie | Wynik | Poprawność wyniku |
|---|---|---|---|
| Klient | `SELECT `IMIE`, `NAZWISKO`, `LOGIN`, `HASLO` FROM `Klient` WHERE `idKlient` = 2;` | 1 row(s) returned | Poprawny |
| Klient | `SELECT * FROM `Zamowienie` WHERE `idKlient` = 2;` | 1 row(s) returned | Poprawny |
| Klient | `SELECT * FROM `Kontakt` WHERE `idKontakt` = 2;` | 1 row(s) returned | Poprawny |
| Klient | `INSERT INTO `Kontakt` (`idKontakt`, `NUMER_TELEFONU`, `EMAIL`) VALUES (6, '887209187', 'white2115@najlepszy.com');` | Error Code: 1142. INSERT command denied to user 'Klient'@'localhost' for table 'kontakt' | Poprawny |
| Klient | `INSERT INTO `Zamowienie` (`idKlient`, `idPojazd`, `STATUS`, `DATA_ZLOZENIA`, `DATA_REALIZACJI`, `CENA`, `CZY_ZAPLACONO`) VALUES (4, 3, 'ZLOZONO', '20.09.10', '20.10.11', 109.29, 'TAK');` | 1 row(s) affected | Poprawny |
| Klient | `SELECT * FROM `Pracownik`;` | Error Code: 1142. SELECT command denied to user 'Klient'@'localhost' for table 'pracownik' | Poprawny |
| Pracownik | `SELECT * FROM `Pojazd`;` | 10 row(s) returned | Poprawny |
| Pracownik | `UPDATE `Raport` SET `idZamowienie` = 3 WHERE `idRaport` = 4;` | 1 row(s) affected Rows matched: 1 Changed: 1 Warnings: 0 | Poprawny |
| Pracownik | `SELECT `idPracownik` FROM `pracownik_pojazd`;` | 26 row(s) returned | Poprawny |
| Pracownik | `DELETE FROM `Kontakty` WHERE `idKontakt` = 1;` | Error Code: 1142. DELETE command denied to user 'Pracownik'@'localhost' for table 'kontakty' | Poprawny |
| Pracownik | `SELECT * FROM `Zamowienie`;` | 8 row(s) returned | Poprawny |
| Pracownik | `SELECT * FROM `Klient`;` | Error Code: 1142. SELECT command denied to user 'Pracownik'@'localhost' for table 'klienci' | Poprawny |
| Kierownik | `DELETE FROM `Klient` WHERE `idKlient` = 5;` | 1 row(s) affected | Poprawny |
| Kierownik | `UPDATE `Kontakt` SET `NUMER_TELEFONU` = '555666321' WHERE `idKontakt` = 8;` | 1 row(s) affected Rows matched: 1 Changed: 1 Warnings: 0 | Poprawny |
| Kierownik | `INSERT INTO `Pracownik` (`idPracownik`, `idKontakt`, `idKierownik`, `IMIE`, `NAZWISKO`, `LOGIN`, `HASLO`, `UPRAWNIENIA`, `PENSJA`, `DATA_ZATRUDNIENIA`)VALUES (1, 1, NULL, 'Janusz', 'Nosal', 'nosaljanek23', 'JANNOS3', 'WLASCICIEL', 10000, '01.01.2009');` | Error Code: 1142. INSERT command denied to user 'Kierownik'@'localhost' for table 'pracownik' | Poprawny |
| Kierownik | `SELECT * FROM `Pojazd`;` | 10 row(s) returned | Poprawny |
| Kierownik | `INSERT INTO `Kontakt` (`idKontakt`, `NUMER_TELEFONU`, `EMAIL`) VALUES (45, '788666555', 'kowalaa@wp.pl');` | 1 row(s) affected | Poprawny |
| Kierownik | `SELECT * FROM `Pracownik`;` | Error Code: 1142. SELECT command denied to user 'Kierownik'@'localhost' for table 'pracownik' | Poprawny |

| | | | |
|---|---|---|---|
| Właściciel | `SELECT * FROM `Pracownik`;` | 14 row(s) returned | Poprawny |
| Właściciel | `DELETE FROM `Klient` WHERE `idKlient` = 2;` | Error Code: 1142. DELETE command denied to user 'Właściciel'@'localhost' for table 'klient' | Poprawny |
| Właściciel | `DELETE FROM `Pracownik` WHERE `idPracownik` = 5;` | 1 row(s) affected | Poprawny |
| Właściciel | `INSERT INTO `zakład_transportowy`.`Pracownik` (`idPracownik`, `idKontakt`, `idKierownik`, `IMIE`, `NAZWISKO`, `LOGIN`, `HASLO`, `UPRAWNIENIA`, `PENSJA`, `DATA_ZATRUDNIENIA`) VALUES (15, 1, NULL, 'Paweł', 'Nowak', 'janek23', 'JANNOS3', 'KIEROWNIK', 10000, '20.12.15');` | 1 row(s) affected | Poprawny |
| Właściciel | `UPDATE `Pojazd` SET `STAN_TECHNICZNY`= 'DO NAPRAWY' WHERE `idPojazd` = 4;` | 1 row(s) affected Rows matched: 1 Changed: 1 Warnings: 0 | Poprawny |
| Właściciel | `SELECT * FROM `Pracownik_pojazd`;` | 19 row(s) returned | Poprawny |

**SPIS TREŚCI**

**SPIS ILUSTRACJI**



**SPIS TABEL**