Politechnika Wrocławska

Wydział Elektroniki

Bazy Danych 2

Laboratorium

System bazodanowy do obsługi hurtowni napojów

Autorzy;
Tomasz Łapszo 226097
Sebastian Jantos 225982

Prowadzący;
dr. inż. Paweł Głuchowski

Wrocław 2017

1. Opracowanie wymagań funkcjonalnych i niefunkcjonalnych dla aplikacji bazodanowej (jak na Inżynierii Oprogramowania).

Wymagania funkcjonalne:

- Monitorowanie ilości produktów w magazynach
- Zamawianie uzupełnień produktów
- Składanie i anulowanie zamówień
- Wykaz zamówień realizowanych oraz stanu ich realizacji
- Generowanie faktur
- Rejestrowanie klientów i dostawców
- Przechowywanie danych pracowników
- Generowanie raportów opisujących obroty hurtowni
- Dodawanie nowych napojów oraz usuwanie istniejących

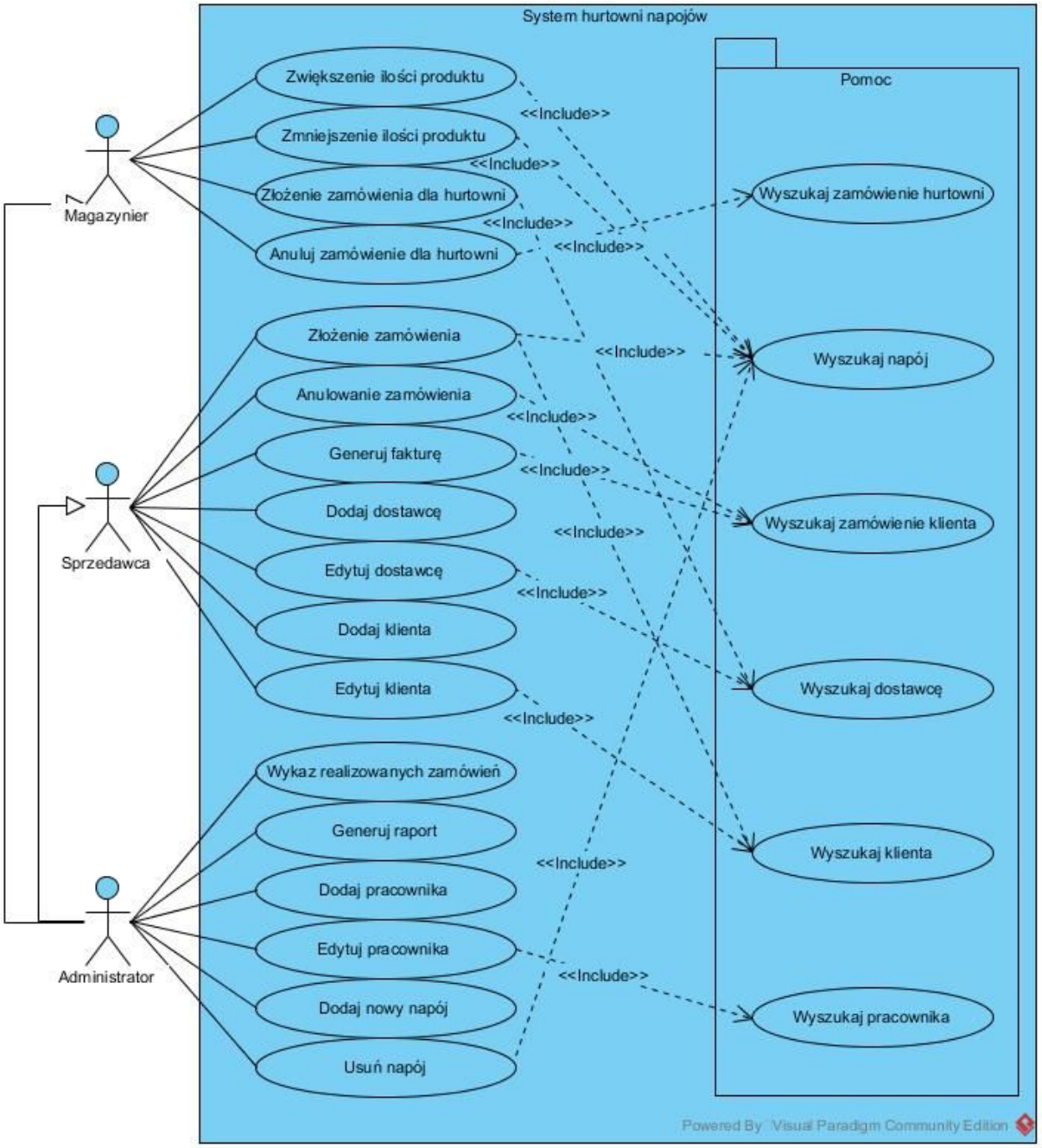
Aplikacja musi także dawać możliwość dodawania, edytowania danych bądź usuwania ich z bazy. Takie uprawnienia posiadać będą tylko uprawnieni użytkownicy.

Wymagania niefunkcjonalne:

- Zapewnienie co najmniej 3 zestawów komputerowych, po jednym dla Magazyniera, Sprzedawcy i Właściciela.
- Serwer, na którym przechowywana będzie baza danych
- Drukarka laserowa do wydruków dokumentów
- Sprzedawca musi mieć telefon do obsługi klientów oraz kasę fiskalną
- Macierz dyskowa wystarczająco duża, by przechowywać kopię zapasową bazy danych
- Wszystkie urządzenia muszą mieć dostęp do szybkiego łącza internetowego pozwalającego na komfortową pracę oraz dostęp do bazy danych
- W celu poprawy niezawodności systemu, zaleca się posiadanie awaryjnego łącza internetowego
- Dostęp do bazy danych możliwy tylko z firmowych komputerów
- Podzielenie magazynu na sektory oznaczone identyfikatorami

2. Specyfikacja wymagań funkcjonalnych za pomocą diagramu przypadków użycia i ich tekstowych opisów (jak na Inżynierii Oprogramowania).

Diagram przypadków użycia:



Opis tekstowy diagramu przypadków użycia:

Definicje aktorów:

PU - przypadek użycia

Aktor	**Opis**	**Przypadki użycia**
Magazynier	Może zarządzać ilością towaru w hurtowni	● PU Zwiększenie ilości produktu powiązane przez <<include>> z PU Wyszukaj napój ● PU Zmniejszenie ilości produktu powiązane przez <<include>> z PU Wyszukaj napój ● PU Złożenie zamówienia dla hurtowni powiązane przez <<include>> z PU Wyszukaj dostawcę ● PU Anuluj zamówienie dla hurtowni powiązane przez <<include>> z PU Wyszukaj zamówienie hurtowni
Sprzedawca	Może dodawać do systemu klientów i dostawców oraz edytować ich dane, tworzyć zamówienia i faktury do niech. Może także anulować zamówienia.	● PU Złożenie zamówienia powiązane przez <<include>> z PU Wyszukaj napój oraz z PU Wyszukaj klienta ● PU Anulowanie zamówienia powiązane przez <<include>> z PU Wyszukaj zamówienie klienta ● PU Generuj fakturę powiązane przez <<include>> z PU Wyszukaj zamówienie klienta ● PU Dodaj dostawcę ● PU Edytuj dostawcę powiązane przez <<include>> z PU Wyszukaj dostawcę ● PU Dodaj klienta ● PU Edytuj klienta powiązane przez <<include>> z PU Wyszukaj klienta
Administrator	Może robić wszystko to, co Magazynier i Sprzedawca, i dodatkowo dodawać pracowników do systemu, edytować ich, określać jakie napoje znajdą się w hurtowni oraz generować raporty.	● PU Wykaz realizowanych zamówień ● PU Generuj raport ● PU Dodaj pracownika ● PU Edytuj pracownika powiązane przez <<include>> z PU Wyszukaj pracownika ● PU Dodaj nowy napój ● PU Usuń napój powiązane przez

		<<include>> z PU Wyszukaj napój

Definicje wymagań funkcjonalnych:

PU Zwiększenie ilości produktu
OPIS
Cel: Zwiększenie ilości produktu w magazynie zapisanej w systemie
Warunki Wstępne: Inicjalizacja w programie (po uruchomieniu)
Warunki Końcowe: Zwiększenie zapisanej ilości napoju
Przebieg: Należy wywołać PU Wyszukaj napój, a następnie zwiększyć ilość zwróconego w trakcie szukania napoju.

PU Zmniejszenie ilości produktu
OPIS
Cel: Zmniejszenie ilości produktu w magazynie zapisanej w systemie
Warunki Wstępne: Inicjalizacja w programie (po uruchomieniu)
Warunki Końcowe: Zmniejszenie zapisanej ilości napoju
Przebieg: Należy wywołać PU Wyszukaj napój, a następnie zmniejszyć ilość zwróconego w trakcie szukania napoju.

PU Złożenie zamówienia dla hurtowni
OPIS
Cel: Zamówienie dostawy napojów, których ilość jest niewystarczająca
Warunki Wstępne: Inicjalizacja w programie (po uruchomieniu)
Warunki Końcowe: Zwiększenie ilości napojów w hurtowni i w systemie (PU Zwiększenie ilości produktu)
Przebieg: Należy wywołać PU Wyszukaj dostawcę, a następnie zamówić odpowiednią ilość towaru. Po dostawie towaru należy użyć PU Zwiększenie ilości produktu, aby odnotować w systemie powiększenie się zapasów.

PU Anulowanie zamówienia dla hurtowni
OPIS
Cel: Anulowanie złożonego zamówienia dla hurtowni
Warunki Wstępne: Inicjalizacja w programie (po uruchomieniu)
Warunki Końcowe: Anulowanie zamówienia
Przebieg: Należy wywołać PU Wyszukaj zamówienie dla hurtowni, a następnie anulować zwrócone zamówienie.

PU Złożenie zamówienia
OPIS
Cel: Sprzedaż napojów klientowi
Warunki Wstępne: Inicjalizacja w programie (po uruchomieniu)
Warunki Końcowe: Zmniejszenie zapisanej ilości napoju oraz odnotowanie zysku
Przebieg: Należy wywołać PU Wyszukaj napój oraz PU Wyszukaj klienta, a następnie utworzyć zamówienie podając odpowiednie dane (dane klienta, jakie napoje i ich ilość).

PU Anulowanie zamówienia
OPIS
Cel: Anulowanie złożonego zamówienia
Warunki Wstępne: Inicjalizacja w programie (po uruchomieniu)
Warunki Końcowe: Anulowanie złożonego zamówienia
Przebieg: Należy wywołać PU Wyszukaj zamówienie klienta, a następnie anulować zwrócone w trakcie szukania zamówienie.

PU Generuj fakturę
OPIS
Cel: Wygenerowanie faktury do zamówienia
Warunki Wstępne: Inicjalizacja w programie (po uruchomieniu)
Warunki Końcowe: Wygenerowanie faktury z danymi zamówienia
Przebieg: Należy wywołać PU Wyszukaj zamówienie klienta, a następnie wygenerować fakturę na podstawie danych z wyszukanego zamówienia.

PU Dodaj dostawcę
OPIS
Cel: Dodanie dostawcy do systemu
Warunki Wstępne: Inicjalizacja w programie (po uruchomieniu)
Warunki Końcowe: Zapisanie danych dostawcy w systemie
Przebieg: Należy dodać dostawcę podając wszystkie jego dane i zapisując je w systemie.

PU Edytuj dostawcę
OPIS
Cel: Zmiana danych dostawcy
Warunki Wstępne: Inicjalizacja w programie (po uruchomieniu)
Warunki Końcowe: Uaktualnienie lub poprawa danych dostawcy
Przebieg: Należy wywołać PU Wyszukaj dostawcę, a następnie edytować dane zwróconego w trakcie szukania dostawcy.

PU Dodaj klienta
OPIS
Cel: Dodanie klienta do systemu
Warunki Wstępne: Inicjalizacja w programie (po uruchomieniu)
Warunki Końcowe: Zapisanie danych klienta w systemie
Przebieg: Należy dodać klienta podając wszystkie niezbędne dane i zapisując je w systemie.

PU Edytuj klienta
OPIS
Cel: Zmiana danych klienta
Warunki Wstępne: Inicjalizacja w programie (po uruchomieniu)
Warunki Końcowe: Uaktualnienie lub poprawa danych klienta zapisanych w systemie
Przebieg: Należy wywołać PU Wyszukaj klienta, a następnie zmienić dane klienta zwrócone w trakcie szukania klienta.

PU Wykaz realizowanych zamówień
OPIS
Cel: Wyświetlenie realizowanych zamówień
Warunki Wstępne: Inicjalizacja w programie (po uruchomieniu)
Warunki Końcowe: Wyświetlenie wszystkich zamówień, których status nie jest oznaczony jako zrealizowane lub anulowane
Przebieg: Należy skorzystać z funkcji systemu generującej wykaz

PU Generuj raport
OPIS
Cel: Wygenerowanie raportu podsumowującego działanie hurtowni
Warunki Wstępne: Inicjalizacja w programie (po uruchomieniu)
Warunki Końcowe: Otrzymanie raportu podsumowującego działanie hurtowni w podanym okresie czasu
Przebieg: Należy skorzystać z funkcji systemu oraz podać okres jaki ma być podsumowany.

PU Dodaj pracownika
OPIS
Cel: Dodanie pracownika do systemu
Warunki Wstępne: Inicjalizacja w programie (po uruchomieniu)
Warunki Końcowe: Zapisanie danych pracownika w systemie
Przebieg: Należy skorzystać z funkcji systemu oraz podać wszystkie wymagane dane pracownika.

PU Edytuj pracownika
OPIS
Cel: Edytowanie pracownika w systemie
Warunki Wstępne: Inicjalizacja w programie (po uruchomieniu)
Warunki Końcowe: Uaktualnienie lub poprawa danych zapisanych w systemie
Przebieg: Należy wywołać PU Wyszukaj pracownika, a następnie zmienić dane zwróconego w trakcie szukania pracownika.

PU Dodaj nowy napój
OPIS
Cel: Dodanie nowego napoju do asortymentu hurtowni
Warunki Wstępne: Inicjalizacja w programie (po uruchomieniu)
Warunki Końcowe: Dodanie nowego napoju do systemu
Przebieg: Należy skorzystać z funkcji systemu oraz podać wszystkie wymagane dane nowego napoju.

PU Usuń napój
OPIS
Cel: Usunięcie napoju z asortymentu hurtowni
Warunki Wstępne: Inicjalizacja w programie (po uruchomieniu)
Warunki Końcowe: usunięcie napoju z systemu hurtowni
Przebieg: Należy wywołać PU Wyszukaj napój, a następnie usunąć zwrócony w trakcie szukania napój.

PU Wyszukaj zamówienie hurtowni
OPIS
Cel: Wyszukanie zamówienia dla hurtowni
Warunki Wstępne: Może być wywołany przez PU Anuluj zamówienie dla hurtowni
Warunki Końcowe: Zwrócenie zamówienia dla hurtowni spełniającego kryteria wyszukiwania
Przebieg: Należy podać kryteria wyszukiwania, a następnie zwrócić pasujące zamówienie jeśli takie jest, jeśli nie, zwrócić informację o braku pasujących zamówień.

PU Wyszukaj napój
OPIS
Cel: Wyszukanie napoju
Warunki Wstępne: Może być wywołany przez PU Zwiększenie ilości produktu, PU Zmniejszenie ilości produktu, PU Usuń napój
Warunki Końcowe: Zwrócenie napoju spełniającego kryteria wyszukiwania
Przebieg: Należy podać kryteria wyszukiwania, a następnie zwrócić pasujący napój jeśli taki jest, jeśli nie, zwrócić informację o braku pasującego napoju.

PU Wyszukaj zamówienie klienta
OPIS
Cel: Wyszukanie zamówienia dla klienta
Warunki Wstępne: Może być wywołany przez PU Anuluj zamówienie, PU Generuj fakturę
Warunki Końcowe: Zwrócenie zamówienia dla klienta spełniającego kryteria wyszukiwania
Przebieg: Należy podać kryteria wyszukiwania, a następnie zwrócić pasujące zamówienie jeśli takie jest, jeśli nie, zwrócić informację o braku pasujących zamówień.

PU Wyszukaj dostawcę
OPIS
Cel: Wyszukanie dostawcy
Warunki Wstępne: Może być wywołany przez PU Edytuj dostawcę
Warunki Końcowe: Zwrócenie dostawcy spełniającego kryteria wyszukiwania
Przebieg: Należy podać kryteria wyszukiwania, a następnie zwrócić pasującego dostawcę jeśli taki jest, jeśli nie, zwrócić informację o braku pasujących dostawców.

PU Wyszukaj klienta
OPIS
Cel: Wyszukanie klienta
Warunki Wstępne: Może być wywołany przez PU Złożenie zamówienia, PU Edytuj klienta
Warunki Końcowe: Zwrócenie klienta spełniającego kryteria wyszukiwania
Przebieg: Należy podać kryteria wyszukiwania, a następnie zwrócić pasującego klienta jeśli taki jest, jeśli nie, zwrócić informację o braku pasujących klientów.

PU Wyszukaj pracownika
OPIS
Cel: Wyszukanie pracownika hurtowni
Warunki Wstępne: Może być wywołany przez PU Edytuj klienta
Warunki Końcowe: Zwrócenie pracownika hurtowni spełniającego kryteria wyszukiwania
Przebieg: Należy podać kryteria wyszukiwania, a następnie zwrócić pasującego klienta jeśli taki jest, jeśli nie, zwrócić informację o braku pasujących pracowników.

3. Identyfikacja encji i opracowanie diagramu związków encji na podstawie analizy scenariuszy przypadków użycia.

Identyfikacja encji:

Definicje typów dla tabeli encji	
Str	Ciąg liter w kodowaniu UNICODE o maksymalnej długości 256 znaków
N	Liczba naturalna
N+	Dodatnia liczba naturalna
R	Nieujemna liczba rzeczywista
R+	Dodatnia liczba rzeczywista
Dat	Data
Bool	Prawda lub fałsz

Klient

Nazwa atrybutu	Opis atrybutu	Typ	OBL (+)
IdKlienta	Numer identyfikacyjny klienta	N+	+
NazwaKlienta	Imię i nazwisko klienta lub nazwa firmy	Str	+
NIP	NIP firmy	N+	+ firma
NrTelefonu	Numer telefonu do klienta	N+	+
Email	Adres email	Str	-
Rabat	Rabat na zakupy	R	+
UlicaNumer	Ulica, numer budynku	Str	+
MiastoKod	Miasto i kod pocztowy klienta	Str	+

Stanowisko

Nazwa atrybutu	Opis atrybutu	Typ	OBL (+)
IdStanowiska	Numer identyfikacyjny stanowiska	N+	+
NazwaStanowiska	Nazwa stanowiska określająca jego uprawnienia	Str	+

Pracownik

Nazwa atrybutu	Opis atrybutu	Typ	OBL (+)
IdPracownika	Numer identyfikacyjny pracownika	N+	+
Nazwisko	Nazwisko pracownika	Str	+
Imię	Imię pracownika	Str	+
NrTelefonu	Numer telefonu do pracownika	N+	+
Email	Adres email	Str	-
UlicaNumer	Ulica, numer budynku	Str	+
MiastoKod	Miasto i kod pocztowy pracownika	Str	+
#IdStanowiska	Numer identyfikacyjny stanowiska	N+	+

Dostawca

Nazwa atrybutu	Opis atrybutu	Typ	OBL (+)
IdDostawcy	Numer identyfikacyjny dostawcy	N+	+
Nazwa dostawcy	Nazwa firmy dostawcy	Str	+
NrTelefonu	Numer telefonu do dostawcy	N+	+
Email	Adres email	Str	-
UlicaNumer	Ulica, numer budynku	Str	+
MiastoKod	Miasto i kod pocztowy dostawcy	Str	+

Napój

Nazwa atrybutu	Opis atrybutu	Typ	OBL (+)
IdNapoju	Numer identyfikacyjny napoju	N+	+
CenaDostawcy	Cena netto dostawcy za sztukę napoju	R+	+
ZawieraCukier	Określa czy napój zawiera cukier	Bool	+
NazwaNapoju	Nazwa napoju	Str	+
#IdRodzajNapoju	Numer identyfikacyjny rodzaju napoju	N+	+
#IdOpakowania	Numer identyfikacyjny opakowania	N+	+
#IdOpakowaniaZbiorczego	Numer identyfikacyjny opakowania zbiorczego	N+	+

Opakowanie

Nazwa atrybutu	Opis atrybutu	Typ	OBL (+)
IdOpakowania	Numer identyfikacyjny opakowania	N+	+
Pojemność	Określa pojemność opakowania	R+	+
Materiał	Określa materiał z jakiego wykonano opakowanie	Str	+

OpakowaniaZbiorcze

Nazwa atrybutu	Opis atrybutu	Typ	OBL (+)
IdOpakowaniaZbiorczego	Numer identyfikacyjny opakowania zbiorczego	N+	+
IleSztuk	Określa ile sztuk jest w opakowaniu zbiorczym	N+	+

RodzajOpakowania	Określa rodzaj opakowania zbiorczego	Str	+
Wymiary	Określa wymiary opakowania zbiorczego	Str	+
SektorMagazynu	Określa sektor magazynu w którym znajduje się opakowanie zbiorcze	Str	+

ZamówieniaHurtowni

Nazwa atrybutu	Opis atrybutu	Typ	OBL (+)
IdZamowieniaHurtowni	Numer identyfikacyjny zamówienia dla hurtowni	N+	+
DataZłożenia	Data złożenia zamówienia	Dat	+
DataPłatności	Data płatności i realizacji zamówienia	Dat	+
ZamówioneAutomatycznie	Określa czy zamówienie zostało złożone automatycznie	Bool	+
#IdStanuZamówienia	Numer identyfikacyjny stanu zamówienia	N+	+
#IdDostawcy	Numer identyfikacyjny dostawcy	N+	+
#IdPracownika	Numer identyfikacyjny pracownika dodającego zamówienie	N+	+

ZamówieniaKlienta

Nazwa atrybutu	Opis atrybutu	Typ	OBL (+)
IdZamówieniaKlienta	Numer identyfikacyjny zamówienia dla klienta	N+	+
DataZłożenia	Data złożenia zamówienia	Dat	+
DataPłatności	Data płatności i realizacji zamówienia	Dat	+
#IdStanuZamówienia	Numer identyfikacyjny stanu zamówienia	N+	+
#IdKlienta	Numer identyfikacyjny klienta	N+	+
#IdPracownika	Numer identyfikacyjny pracownika dodającego zamówienie	N+	+

StanyZamówienia

Nazwa atrybutu	Opis atrybutu	Typ	OBL (+)
IdStanuZamówienia	Numer identyfikacyjny stanu zamówienia	N+	+
NazwaStanu	Stan zamówienia	Str	+

ZamawianeProduktyHurtowni

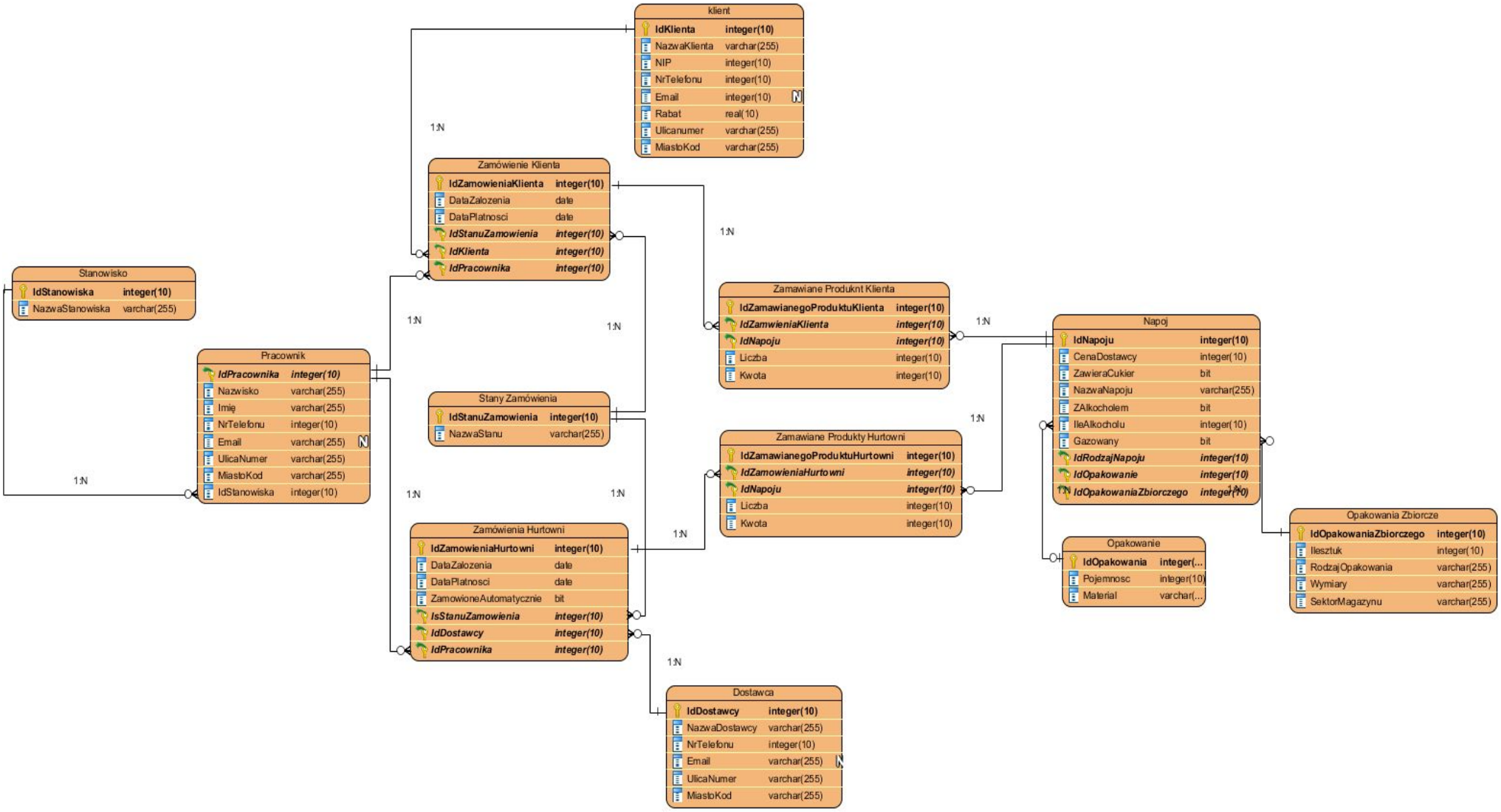
Nazwa atrybutu	Opis atrybutu	Typ	OBL (+)
IdZamawianegoProduktuHurtowni	Numer identyfikacyjny zamawianego dla hurtowni produktu	N+	+
#IdZamówieniaHurtowni	Numer identyfikacyjny zamówienia dla hurtowni	N+	+
#IdNapoju	Numer identyfikacyjny napoju	N+	+
Liczba	Ilość zamówionego napoju	N+	+
Kwota	Koszt zamówienia	R+	+

ZamawianeProduktyKlienta

Nazwa atrybutu	Opis atrybutu	Typ	OBL (+)
IdZamawianegoProduktuKlienta	Numer identyfikacyjny zamawianego dla klienta produktu	N+	+
#IdZamówieniaKlienta	Numer identyfikacyjny zamówienia dla klienta	N+	+
#IdNapoju	Numer identyfikacyjny napoju	N+	+
Liczba	Ilość zamówionego napoju	N+	+
Kwota	Koszt zamówienia	R+	+

Diagram związków encji:

Definicje typów dla diagramu encji	
Varchar(255)	Ciąg liter w kodowaniu UNICODE o maksymalnej długości 256 znaków
Integer	Dodatnia liczba naturalna
R	Nieujemna liczba rzeczywista
Date	Data
Bit	Prawda lub fałsz


klient
IdKlienta integer(10)
NazwaKlienta varchar(255)
NIP integer(10)
NrTelefonu integer(10)
Email integer(10)
Rabat real(10)
Ulicanumer varchar(255)
MiastoKod varchar(255)
Zamówienie Klienta
IdZamowieniaKlienta integer(10)
DataZalozenia date
DataPlatnosci date
IdStanuZamowienia integer(10)
IdKlienta integer(10)
IdPracownika integer(10)
Stanowisko
IdStanowiska integer(10)
NazwaStanowiska varchar(255)
Pracownik
IdPracownika integer(10)
Nazwisko varchar(255)
Imię varchar(255)
NrTelefonu integer(10)
Email varchar(255)
UlicaNumer varchar(255)
MiastoKod varchar(255)
IdStanowiska integer(10)
Stany Zamówienia
IdStanuZamowienia integer(10)
NazwaStanu varchar(255)
Zamawiane Produknt Klienta
IdZamawianegoProduktuKlienta integer(10)
IdZamwieniaKlienta integer(10)
IdNapoju integer(10)
Liczba integer(10)
Kwota integer(10)
Napoj
IdNapoju integer(10)
CenaDostawcy integer(10)
ZawieraCukier bit
NazwaNapoju varchar(255)
ZAlkocholem bit
IleAlkocholu integer(10)
Gazowany bit
IdRodzajNapoju integer(10)
IdOpakowanie integer(10)
IdOpakowaniaZbiorczego integer(10)
Zamawiane Produkty Hurtowni
IdZamawianegoProduktuHurtowni integer(10)
IdZamowieniaHurtowni integer(10)
IdNapoju integer(10)
Liczba integer(10)
Kwota integer(10)
Opakowania Zbiorcze
IdOpakowaniaZbiorczego integer(10)
Ilesztuk integer(10)
RodzajOpakowania varchar(255)
Wymiary varchar(255)
SektorMagazynu varchar(255)
Opakowanie
IdOpakowania integer(...
Pojemnosc integer(10)
Material varchar(...
Zamówienia Hurtowni
IdZamowieniaHurtowni integer(10)
DataZalozenia date
DataPlatnosci date
ZamowioneAutomatycznie bit
IsStanuZamowienia integer(10)
IdDostawcy integer(10)
IdPracownika integer(10)
Dostawca
IdDostawcy integer(10)
NazwaDostawcy varchar(255)
NrTelefonu integer(10)
Email varchar(255)
UlicaNumer varchar(255)
MiastoKod varchar(255)
1:N

4. Analiza liczby instancji dla każdej encji.

Nazwa encji	Średnia liczba instancji	Maksymalna liczba instancji
Klient	1 000	5 000
Stanowisko	3	5
Pracownik	10	20
Dostawca	50	100
Napój	1 000	3 000
Opakowanie	3	12
Opakowanie zbiorcze	2	5
Zamówienie hurtowni	1000	10 000+
Zamówienie klienta	1000	10 000+
Stan zamówienia	4	4
Zamawiane produkty hurtowni	2000	10 000+
Zamawiane produkty klienta	2000	10 000+

5. Analiza użycia encji, identyfikująca podstawowe rodzaje transakcji: wstawianie, modyfikacja usuwanie i wyszukiwanie. Określenie na tej podstawie zmienności zawartości poszczególnych tabel.

Podstawowe transakcje

1. Wstawianie:
    1.1. Złożenie zamówienie dla hurtowni
    1.2. Dodanie zamawianego produktu hurtowni
    1.3. Dodanie zamawianego produktu klienta
    1.4. Złożenie zamówienia
    1.5. Dodanie dostawcy
    1.6. Dodanie klienta
    1.7. Dodanie pracownika
    1.8. Dodanie nowego napoju

2. Modyfikacja:
    2.1. Zwiększenie ilości produktu
    2.2. Zmniejszenie ilości produktu
    2.3. Edytowanie dostawcy
    2.4. Edytowanie klienta
    2.5. Edytowanie pracownika
    2.6. Edytowanie napoju

3. Usuwanie:
    3.1. Anulowanie zamówienie dla hurtowni
    3.2. Anulowanie zamówienia
    3.3. Usuwanie napoju

4. Wyszukiwanie:
    4.1. Generowanie faktury
    4.2. Wykaz realizowanych zamówień
    4.3. Generowanie raportu
    4.4. Wyszukanie zamówienia hurtowni (data złożenia, dostawcy lub pracowniku)
    4.5. Wyszukanie napoju
    4.6. Wyszukanie zamówienia klienta
    4.7. Wyszukanie dostawcy
    4.8. Wyszukanie klienta
    4.9. Wyszukanie pracownika

Nazwa tabeli	**Zmienność**
Klienci	Mała zmienność
Stanowiska	Mała zmienność
Pracownicy	Mała zmienność
Dostawcy	Duża zmienność
Napoje	Duża zmienność
Opakowania	Mała zmienność
Opakowania zbiorcze	Mała zmienność
Zamówienia hurtowni	Duża zmienność
Zamówienia klienta	Duża zmienność
Stany zamówienia	Mała zmienność
Zamawiane produkty hurtowni	Mała zmienność
Zamawiane produkty klientów	Mała zmienność

6. Sformułowanie wymagań dotyczących dostępu i określenie częstości wykonywania operacji na danych np. tworzenia raportów.

Encja	Grupa dostępu	Częstotliwość wykonywania transakcji	Częstotliwość w liczbach
Stanowisko	Magazynier, Sprzedawca, Administrator	Encja “Stanowisko” będzie nie zmieniana, gdyż służy ona tylko do określenia dostępu do poszczególnych elementów programu bazodanowego.	Stanowiska uzupełnianie raz na początku funkcjonowania bazy danych; 4 operacje.
Klient	Sprzedawca, Administrator	Encja “ Klient” będzie najbardziej obciążona na początku powstania hurtowni, gdyż będzie wtedy najwięcej nowych klientów. Z czasem będą dochodzić nowi ale będzie to odbywać się w założeniach 2 na tydzień.	Na początku istnienia hurtowni do około 1000 operacji, potem 2 operacje na tydzień.

Pracownik	Administrator, Sprzedawca, Magazynier	Operacje na tej encji wykonywane będą bardzo rzadko. Skład pracowników nie zmienia się często, a zapotrzebowanie na nowych pracowników jest znikome. Największe natężenie będzie na początku istnienia hurtowni.	Przez pierwszy miesiąc do 50 operacji, potem 2-3 operacje na rok.
Dostawca	Administrator, Magazynier	Podobnie do encji “ Pracownik”. Na początku istnienia hurtowni zostaną dodani do listy wcześniej wybrani dostawcy, z czasem będą dodawani nowi, jeżeli któryś z poprzednich odmówił współpracy.	Przez pierwszy miesiąc do 50 operacji, potem 2-3 operacje na rok.
Napój	Administrator	Encja “ Napój” będzie zmieniać się często ze względu na zmianę produktów przez producentów.	Na początku istnienia bazy do 1000 operacji, potem 10 na miesiąc.
Opakowanie	Administrator	Napoje są pakowane w standardowe opakowania, więc ich zbiór został wprowadzony do tworzonej bazy i nie będzie zmieniany.	Około 50 operacji przy tworzeniu bazy.
Opakowanie zbiorcze	Administrator	Analogicznie do opakowania napoju, zbiorcze opakowania są definiowane przy tworzeniu bazy i nie zmieniane.	Około 50 operacji przy tworzeniu bazy.
Zamówienia Hurtowni	Administrator, Magazynier	Zmienność zależy od encji “Zamówienia Klienta”, ponieważ zamówienia hurtowni będą składane tylko w przypadkach kończącego się towaru.	Co miesiąc około 200 zamówień.
Zamówienia Klienta	Administrator, Sprzedawca	Encja “ Zamówienie Klienta” będzie się zmieniała w zależności od zapotrzebowania na produkty przez klientów. Najwięcej będzie ich w okresach przedświątecznych, gdzie sklepy potrzebują więcej towarów. Jest to encja o największej ilości operacji.	Co miesiąc około 200 zamówień.

Stan zamówienia	Administrator	Stany zamówienia będą zdefiniowane na samym początku istnienia hurtowni i nie będą zmieniane w przyszłości.	4 operacje w 1 dniu istnienia bazy.
Zamawiane ProduktyHurtowni	Administrator, Magazynier	Każde zamówienie hurtowni może zawierać wiele produktów, więc na tej encji operacje będa wykonywane bardzo często.	Co miesiąc około 400 zamówień.
Zamawiane ProduktyKlienta	Administrator, Sprzedawca	Analogicznie do encji "Zamawiane Produkty Hurtowni".	Co miesiąc około 400 zamówień.

7. Analiza i poprawa integralności.

Już na etapie projektowania, systemu bazodanowego, zwracano uwagę na relacje między tabelami, które mogą być potencjalnym źródłem problemów w zachowaniu integralności danych bazy. Ograniczono je do minimum, dzięki czemu zmniejszono ilość błędów mogących powstać w bazie danych. Także aplikacja pozwalająca na wykonywanie pewnych operacji na danych przechowywanych w bazie, ma z góry ustalone funkcje, których użytkownik potrzebuje aby baza była funkcjonalna. Dzięki temu można określić skończony zbiór operacji na bazie danych i wśród nich znaleźć te, które mogą powodować utratę integralności, a następnie wprowadzenie kontroli nad nimi. Pozwoli to na zachowanie integralności.

Jednym z głównych zadań systemu będzie kontrolowanie wprowadzanych przez użytkownika danych, aby w bazie nie pojawiły się niedozwolone wartości.

8. Dostrajanie bazy danych pod względem wydajności (na podst. 1–4):

– tworzenie mechanizmów dostępu do bazy i danych (m.in. procedur),

Aby zwiększyć bezpieczeństwo integralności bazy danych oraz ograniczyć obciążenie serwera stworzone zostaną procedury obsługujące operacje dodawania, modyfikacji oraz usuwania danych. Procedury są przechowywane na serwerze bazy danych, więc korzystanie z nich zapewni brak nieautoryzowanych zmian w tabelach bazy. Wyszukiwania nie mają wpływu na integralność bazy danych, ponieważ nie ingerują w przechowywane dane, jednak zostanie stworzona procedura wyszukująca, która poprawi wydajność zapytań.

– dodawanie indeksów (tam, gdzie poprawią działanie bazy),

W projektowanej bazie danych dwoma największymi tabelami będą Klienci oraz Napoje. Obie tabele mogą zawierać około tysiąca rekordów. Są to też tabele, na których operacje będą wykonywane najczęściej, więc wymagany jest szybki dostęp do danych w nich zawartych. Umożliwi to dodanie indeksowania klucza głównego, znacznie przyspieszy to operacje wyszukiwania danych, kosztem zwiększenia potrzebnej pamięci dyskowej. Jest to jednak opłacalne.

– denormalizacja (jeśli poprawi działanie bazy),

Zdecydowano się na połączenie encji “Napój” i “Rodzaj napoju”, ponieważ w zdecydowanej większości przypadków mamy do czynienia z relacją 1;1, co zajmuje więcej pamięci, spowalnia wykonywania operacji i utrudnia dostęp do systemu.

– obsługa więzów integralności.

Operacje mające wpływ na inne tabele:

- **Złożenie zamówienia**
  W przypadku, gdy klient składa zamówienie, już w momencie jego złożenia, w magazynie trzeba odnotować zmniejszenie ilości towaru. Zatem nowe zamówienie powoduje zmianę wartości pola “Magazynowana ilość” w tabeli Napoje. Dodatkowo złożenie zamówienia powoduje dodanie nowego rekordu w tabeli Zamawiane produkty klientów.
- **Anulowanie zamówienia**
  W przypadku anulowania zamówienia, należy zwiększyć ilość magazynowanego towaru. Anulowanie zamówienia zmienia wartość pola “Magazynowana ilość” w tabeli Napoje. Ta operacja skutkuje też usunięciem odpowiadającego jej rekordu z tabeli Zamawiane produkty klientów.
- Analogicznie do powyższych operacji **złożenie zamówienia hurtowni i jego anulowanie** dodaje lub usuwa rekord z tabeli Zamawiane produkty hurtowni.
- Operacja **usunięcia napoju** z hurtowni musi sprawdzić, czy usuwany napój nie znajduje się w którymkolwiek z aktualnych zamówień klientów, bądź zamówień hurtowni. Dopiero gdy napój nie występuje w żadnym zamówieniu, można go usunąć

9. Fizyczny projekt bazy danych.

Baza danych została zaimplementowana w języku MySQL w środowisku phpmyadmin. Środowisko phpmyadmin jest uruchamiane za pomocą programu xampp na localhost. W pierwszym kroku stworzono bazę oraz dodano tabele zgodnie z diagramem encji znajdującym się w punkcie 3. Do bazy dodano kilka rekordów w celu przetestowania jej działania, a następnie stworzono procedury, które pokryją wszystkie funkcjonalności aplikacji. Tworzenie bazy, polega na wykonaniu skryptu tworzącego bazę, podanego w dodatku .1

10. Zbiór zapytań zoptymalizowanych.

Procedura DodajKlienta
parametry: nazwa klienta, NIP, nr. telefonu, email, rabat, ulica i numer, miasto i kod.
Uwagi:

- Rabat jest liczbą naturalną z zakresu [0, 100], która wyraża procentowy upust cen.
- Email nie jest obowiązkowym parametrem.

Procedura DodajPracownika
parametry: nazwisko, imię, nr. telefonu, email, ulica i numer, miasto i kod, ID stanowiska.
Uwagi:

- Email nie jest obowiązkowym parametrem.

Procedura DodajDostawce
parametry: nazwa dostawcy, nr. telefonu, email, ulica i numer, miasto i kod
Uwagi:

- Email nie jest obowiązkowym parametrem.

Procedura DodajNapoj
parametry: nazwa napoju, cena dostawcy, zawiera cukier, z alkoholem, ile alkoholu, gazowany, sektor magazynu, stan magazynu, ID opakowania, ID opakowania zbiorczego.
Uwagi

- Cena dostawcy to cena netto napoju, separator dziesiętny to kropka “.”.
- Zawiera cukier, z alkoholem, gazowany to wartości TRUE/FALSE.
- Sektor magazynu to oznaczenie przyjęte w danym magazynie.

Procedura ZlozZamowienieKlienta
parametry: data złożenia, data płatności, ID stanu zamówienia, ID klienta, ID pracownika.
Uwagi

- Data złożenia oraz data płatności to daty podane w formacie DD/MM/RRRR.

Procedura ZlozZamowienieHurtowni
parametry: data złożenia, data płatności, zamówienie automatyczne, ID stanu zamówienia, ID dostawcy, ID pracownika.
Uwagi

- Data złożenia oraz data płatności to daty podane w formacie DD/MM/RRRR.
- Zamówienie automatyczne to wartość TRUE/FALSE.

Procedura DodajZamawianyProduktHurtowni
parametry: ID zamówienia hurtowni, ID napoju, zamówiona ilość.
Uwagi

- Należność za daną ilość zamawianego produktu jest obliczana jako zamówiona ilość * cena dostawcy.

Procedura DodajZamawianyProduktKlienta
parametry: ID zamówienia klienta, ID napoju, zamówiona ilość.
Uwagi

- Należność za daną ilość zamawianego produktu jest obliczana jako zamówiona ilość * 123% * cena dostawcy.

Procedura UsunNapoj
parametry: id.
Uwagi

- Parametr id jest kluczem głównym napojów.

Procedura AnulujZamowienieKlienta
parametry: id.
Uwagi

- Parametr id jest kluczem głównym zamówień klientów.
- Wszystkie zamawiane napoje powiązane z tym zamówieniem zostaną usunięte.
- Zamówienie nie jest usuwane lecz przechodzi do historii ze stanem “anulowane”.

Procedura AnulujZamowienieHurtowni
parametry: id.
Uwagi

- Parametr id jest kluczem głównym zamówień hurtowni
- Wszystkie zamawiane napoje powiązane z tym zamówieniem zostaną usunięte.
- Zamówienie nie jest usuwane lecz przechodzi do historii ze stanem “anulowane”.

Procedura ZmienIloscProduktu
parametry: ID napoju, nowa ilość
Uwagi

- Nowa ilość powinna uwzględniać stan magazynu (zastępuje stan magazynu a nie zwiększa czy obniża).

Procedura EdytujDostawce
parametry: ID dostawcy, nazwa dostawcy, nr. telefonu, email, ulica i numer, miasto i kod.
Uwagi

- Email nie jest obowiązkowym parametrem.
- Zmianie ulegają tylko wybrane parametry, reszta będzie przekazana oryginalna.

Procedura EdytujKlienta
parametry: ID klienta, nazwa klienta, NIP, nr. telefonu, email, rabat, ulica i numer, miasto i kod.
Uwagi

- Email nie jest obowiązkowym parametrem.
- Zmianie ulegają tylko wybrane parametry, reszta będzie przekazana oryginalna.

Procedura EdytujPracownika
parametry: ID pracownika, nazwisko, imię, , nr. telefonu, email, ulica i numer, miasto i kod, ID stanowiska.
Uwagi

- Email nie jest obowiązkowym parametrem.
- Zmianie ulegają tylko wybrane parametry, reszta będzie przekazana oryginalna.

Procedura EdytujZamawianyProduktHurtowni
parametry: ID zamawianego produktu, ID zamówienia hurtowni, ID napoju, ilość.
Uwagi

- Należność za daną ilość zamawianego produktu jest obliczana jako zamówiona ilość * cena dostawcy.

Procedura EdytujZamawianyProduktKlienta
parametry: ID zamawianego produktu, ID zamówienia klienta, ID napoju, zamówiona ilość.
Uwagi

- Należność za daną ilość zamawianego produktu jest obliczana jako zamówiona ilość * 123% * cena dostawcy.
-

11. Zasady polityki bezpieczeństwa.

1. Magazynier nie ma dostępu do funkcji Administratora i Sprzedawcy.
2. Sprzedawca nie ma dostępu do funkcji Administratora i Magazyniera.
3. Administrator ma dostęp do wszystkich funkcji bazy.
4. System bazodanowy posiada kopie zapasową w której archiwizuje wszystkie zamówienia hurtowni, zamówienia klienta, zamawiane produkty klienta oraz zamawiane produkty hurtowni.
5. Wszystkie identyfikatory są liczbami naturalnymi numerowanymi w kolejności dodania wpisu do tabeli.
6. Zarówno numer budynku jak i numer lokalu mogą być podawane, jako ciągi znaków (niekoniecznie samych cyfr), ponieważ występuje wiele przypadków, kiedy w skład numeru wchodzi litera.
7. Dla klienta będącego osobą fizyczną wymagane jest podanie imienia i nazwiska, natomiast w przypadku, gdy klientem jest firma, wymaga się podania nazwy firmy oraz NIP-u.
8. Adres e-mail pasuje do następującego wyrażenia regularnego: „^[a-zA-Z0-9_.+-]+@[a-zA-Z0-9-]+\.[a-zA-Z0-9-.]+$”.
9. Stawka podatku, zawartość alkoholu oraz rabat klienta zapisywane są jako dodatnia i rzeczywista wartość liczbowa z symbolem „%”, np. 1%.
10. Numer telefonu nie jest poprzedzony numerem kierunkowym.
11. 8.Format każdej daty to DD/MM/RRRR, gdzie DD - dzień, MM - miesiąc, RRRR - rok, np. 01/01/2001
12. Data ważności produktu nie może być wcześniejsza niż data jego zamówienia.
13. Nazwa smaku może występować jako ciąg znaków oddzielony myślnikami np. banan-marchew-jabłko
14. Kwota końcowa zamówienia klienta obliczana jest na sumy iloczynu liczby zakupionych produktów i ich ceny za jedną sztukę.
15. Dostępna liczba sztuk napoju musi być nie większa niż jego liczba sztuk.
16. Liczba sztuk napoju, dostępna liczba sztuk napoju oraz minimalna liczba sztuk napoju muszą być liczbami naturalnymi.
17. Cena hurtowni dla wybranego napoju nie może być niższa niż cena napoju u producenta.
18. Ceny występują jako dodatnie liczby rzeczywiste wraz z symbolem waluty np. 3 zł
19. Wymiary opakowania zbiorczego podaje się jako ciąg znaków : XxYxZ, gdzie X,Y,Z to dodanie liczby całkowite wyrażane jako milimetry odpowiednio :
    X - wysokość,
    Y - szerokość,
    Z - głębokość.
20. Każde zamówienie klienta dotyczy tylko jednego klienta.
21. Nie może istnieć zamówienie klienta, które nie dotyczy żadnego klienta.

22. Klient nie musi być powiązany z żadnym zamówieniem klienta.
23. Każde zamówienie klienta musi posiadać stan realizacji zamówienia.
24. Nie może istnieć zamówienie klienta, które nie posiada stanu realizacji zamówienia.
25. Wiele zamówień klienta może znajdować się w tym samym stanie realizacji zamówienia.
26. Stan zamówienia nie musi być powiązany z żadnym zamówieniem klienta.
27. Każde zamówienie hurtowni musi posiadać stan realizacji zamówienia.
28. Nie może istnieć zamówienie hurtowni, które nie posiada stanu realizacji zamówienia
29. Wiele zamówień hurtowni może znajdować się w tym samym stanie realizacji zamówienia
30. Stan zamówienia nie musi być powiązany z żadnym zamówieniem hurtowni.
31. Zamówienie klienta musi mieć określoną jedną formę płatności.
32. Nie może istnieć zamówienie klienta, które nie posiada określonej formy płatności.
33. Może istnieć forma płatności, która nie dotyczy żadnego zamówienie klienta.
34. Zamówienie hurtowni musi mieć tą samą formę płatności.
35. Nie może istnieć zamówienie hurtowni, które nie posiada określonej formy płatności.
36. Może istnieć forma płatności, która nie dotyczy żadnego zamówienie hurtowni.
37. Zamówienie hurtowni musi dotyczyć jednego dostawcy.
38. Nie może istnieć zamówienie hurtowni, które nie jest powiązane z żadnym dostawcą.
39. Jeden dostawca może dostarczać wiele zamówień hurtowni.
40. Może istnieć dostawca, który nie dostarcza żadnego zamówienia hurtowni.
41. Napój dostawcy musi być powiązany tylko z jednym dostawcą.
42. Nie może istnieć napój dostawcy, który nie jest powiązany z żadnym dostawcą.
43. Dostawca może dostarczać wiele napojów dostawcy.
44. Może istnieć dostawca, który nie dostarcza żadnego napoju dostawcy.
45. Pracownik w imieniu klienta może złożyć wiele zamówień klienta.
46. Może istnieć pracownik, który nigdy nie składał w imieniu klienta żadnego zamówienia klienta.
47. Zamówienie klienta może być złożone tylko przez jednego pracownika.
48. Nie może istnieć zamówienie klienta, które nie zostało złożone przez żadnego pracownika.
49. Pracownik może złożyć wiele zamówień hurtowni.
50. Może istnieć pracownik, który nigdy nie składał żadnego zamówienia hurtowni.
51. Zamówienie hurtowni może być złożone tylko przez jednego pracownika.
52. Nie może istnieć zamówienie hurtowni, które nie zostało złożone przez żadnego pracownika.
53. Każdy pracownik musi mieć przypisane jedno stanowisko.
54. Nie może istnieć pracownik, który nie ma przypisanego żadnego stanowiska.
55. Na jednym stanowisku może pracować wielu pracowników.
56. Może istnieć stanowisko, do którego nie jest przypisany żaden pracownik.
57. Zamawiany produkt klienta musi dotyczyć jednego zamówienia klienta.

58. Nie może istnieć zamawiany produkt klienta, który nie dotyczy żadnego zamówienia klienta.
59. Może istnieć zamówienie klienta, które nie posiada jeszcze żadnych zamawianych produktów.
60. Zamawiany produkt hurtowni musi dotyczyć jednego zamówienia hurtowni.
61. Nie może istnieć zamawiany produkt hurtowni, który nie dotyczy żadnego zamówienia hurtowni.
62. Może istnieć zamówienie hurtowni, które nie posiada jeszcze żadnych zamawianych produktów.
63. Jedno zamówienie hurtowni może dotyczyć wielu zamawianych produktów hurtowni.
64. Każdy napój dostawcy może być powiązany tylko z jednym napojem.
65. Nie może istnieć napój dostawcy, który nie jest powiązany z żadnym napojem.
66. Jeden napój może dotyczyć wielu pozycji napojów dostawcy.
67. Może istnieć napój, który nie dotyczy żadnej pozycji napoju dostawcy.
68. Każdy zamawiany produkt klienta może dotyczyć tylko jednego napoju.
69. Nie może istnieć zamawiany produkt klienta, który nie jest powiązany z żadnym napojem.
70. Jeden napój może dotyczyć wielu pozycji zamawianych produktów klienta.
71. Napój nie musi dotyczy żadnej pozycji zamawianych produktów klienta.
72. Każdy zamawiany produkt hurtowni może dotyczyć tylko jednego napoju.
73. Nie może istnieć zamawiany produkt hurtowni, który nie jest powiązany z żadnym napojem.
74. Jeden napój może dotyczyć wielu pozycji zamawianych produktów hurtowni.
75. Napój nie musi dotyczy żadnej pozycji zamawianych produktów hurtowni.
76. Jedna pozycja magazynu może przechowywać tylko jeden napój.
77. Nie może istnieć pozycja magazynu, która nie przechowuje informacji o żadnym napoju.
78. Każdy napój musi posiadać tylko jednego producenta.
79. Nie może istnieć napój, który nie jest powiązany z żadnym producentem.
80. Jeden producent może być powiązany z wieloma napojami.
81. Może istnieć producent, który nie jest powiązany z żadnym napojem.
82. Każdy napój może zawierać tylko jedną nazwę napoju.
83. Nie może istnieć napój, który nie posiada nazwy napoju.
84. Jedną nazwę napoju może posiadać wiele napojów.
85. Może istnieć nazwa napoju, której nie posiada żaden napój.
86. Każdy napój musi zawierać smak.
87. Nie może istnieć napój, który nie posiada smaku.
88. Jeden smak może posiadać wiele napojów.
89. Może istnieć smak, której nie posiada żaden napój.
90. Każdy napój musi zawierać określony rodzaj gazu.
91. Nie może istnieć napój, który nie ma określonego rodzaj gazu.
92. Jeden rodzaj gazu może posiadać wiele napojów.

93. Może istnieć rodzaj gazu, który nie dotyczy żadnego napoju.
94. Każdy napój musi mieć określony rodzaj napoju.
95. Nie może istnieć napój, który nie ma określonego rodzaj napoju.
96. Jeden rodzaj napoju może posiadać wiele napojów.
97. Może istnieć rodzaj napoju, który nie dotyczy żadnego napoju.
98. Każdy napój musi mieć określony rodzaj opakowania zbiorczego.
99. Nie może istnieć napój, który nie ma określonego rodzaju opakowania zbiorczego.
100. Jeden rodzaj opakowania zbiorczego, może posiadać wiele napojów.
101. Może istnieć rodzaj opakowania zbiorczego, który nie dotyczy żadnego napoju.
102. Każdy rodzaj opakowania musi posiadać jedną pojemność opakowania.
103. Nie może istnieć opakowanie, które nie ma pojemności opakowania.
104. Jedną pojemności opakowania może posiadać wiele rodzajów opakowań.
105. Może istnieć pojemność opakowania, który nie dotyczy żadnego opakowania.
106. Każdy rodzaj opakowania zbiorczego musi posiadać wymiary opakowania.
107. Nie może istnieć opakowanie zbiorcze, które nie ma wymiarów opakowania.
108. Jedną wymiar opakowania może posiadać wiele rodzajów opakowań zbiorczych.
109. Może istnieć wymiar opakowania, który nie dotyczy żadnego opakowania zbiorczego
110. Każdy rodzaj opakowania zbiorczego musi posiadać jeden rodzaj opakowania.
111. Nie może istnieć opakowanie zbiorcze, które nie ma rodzaju opakowania.
112. Jeden rodzaj opakowania może posiadać wiele rodzajów opakowań zbiorczych.
113. Może istnieć rodzaj opakowania, który nie dotyczy żadnego opakowania zbiorczego.
114. Każdy napoju musi zawierać jedną nazwę rodzaju napoju.
115. Nie może istnieć napoju, który nie ma określonej nazwy napoju.
116. Jedna nazwa napoju może dotyczyć wielu rodzajów napojów.
117. Każdy napoju musi zawierać jedną zawartość alkoholu.
118. Nie może istnieć napoju, który nie ma określonej zawartości alkoholu.
119. Jedna zawartość alkoholu może dotyczyć wielu napojów.
120. Może istnieć zawartość alkoholu, która nie dotyczy żadnego napoju. /121
121. Magazynier może aktualizować dane towarów.
122. W bazie mogą istnieć dane towaru, który jeszcze nie został sprzedany
123. Wszystkie dotyczące zrealizowanych zamówień są archiwizowane, nie podlegają usunięciu przez okres 5-ciu lat.
124. Każdy Klient może kupić w jednym zamówieniu wiele towarów.
125. Jeden towar może być kupiony w różnych ilościach przez różnych Klientów.
126. Klient może zrezygnować z zamówienia, dopóki zamówienie jest w stanie niezrealizowanym.
127. Dane Klienta wprowadza Sprzedawca lub Administrator.
128. Dane Klienta mogą być modyfikowane przez Sprzedawcę lub Administratora.
129. Dane Klienta mogą być usuwane przez Administratora.
130. Dane Dostawcy wprowadza Administrator i Sprzedawcę.

131. Dane Dostawcy mogą być modyfikowane przez Administratora i Sprzedawcę.
132. Dane Zamówienia klienta wprowadza Sprzedawca lub Administrator.
133. Dane Zamówienia hurtowni wprowadza Magazynier lub Administrator.
134. Dane Pracowników mogą być usuwane przez Administratora.
135. Dane Produktów mogą być usuwane przez Administratora.
136. Zamówienia klienta mogą być modyfikowane przez Sprzedawcę i Administratora.
137. Administrator może modyfikować dane produktów.
138. Kwota zamówienia jest automatycznie wyliczana na podstawie zamówionych produktów.
139. Administrator otrzymuje informacje o towarach od przedstawicieli handlowych i dostawców, które wprowadza do bazy danych.
140. Sprzedawca może na prośbę Klienta złożyć wiele zamówień Klienta.
141. Jedno zamówienie na prośbę Klienta może złożyć tylko jeden pracownik (Sprzedawca).
142. Sprzedawca może w imieniu klienta edytować jego zamówienie.
143. Zamówienie klienta składane przez Sprzedawcę muszą dotyczyć Klienta, który zleca takie zamówienie.
144. System automatycznie generuje zamówienie hurtowni, jeśli liczba sztuk w pozycji magazynu jest równa 0 lub jest niższa niż ustalona minimalna liczba sztuk.
145. Nie można modyfikować zrealizowanych zamówień klienta i zamówień hurtowni.

12. *Implementacja i testy aplikacji bazodanowej*

Implementacja aplikacji zawiera wszystkie kluczowe funkcje, które są potrzebne by móc działać z zaprojektowaną bazą danych. Nastąpiły pewne drobne zmiany, np. usunięcie daty płatności z tabeli zamówień. Dodatkowo aplikacja nie przewiduje możliwości zmiany stanu zamówienia, ponieważ docelowo, powinno się to dziać automatycznie, jednak wymagałoby to implementacji co najmniej jednego dodatkowego systemu. Możliwa jest jedynie zmiana stanu na anulowany, ponieważ ta zmiana jest możliwa do zrobienia przez uprawnionego pracownika hurtowni. W zamieszczonym w tym dokumencie diagramie opisujących bazę danych nie ma pól loginu i hasła w tabeli pracownicy, zostały one dodane później, gdy implementowano logowanie się pracowników. Oba dodane pola przechowują jedynie VARCHAR(128) czyli wygenerowany za pomocą SHA512 hash. Nie udało się również zaimplementować generowania raportu i danych do rachunku.

Dodatek 1

Skrypt generujący bazę danych z kilkoma testowymi rekordami i procedury

```
CREATE DATABASE warehouse;

USE warehouse;

CREATE TABLE warehouse.clients(
   ID_client MEDIUMINT UNSIGNED NOT NULL AUTO_INCREMENT PRIMARY KEY,
   clientName VARCHAR(100) NOT NULL,
   nip INT(10) UNSIGNED NOT NULL,
   phoneNumber int(9) UNSIGNED NOT NULL,
   email VARCHAR(50),
   discount INT(3) NOT NULL,
   street VARCHAR(50) NOT NULL,
   city VARCHAR(50) NOT NULL);

CREATE TABLE warehouse.positions(
   ID_position MEDIUMINT UNSIGNED NOT NULL AUTO_INCREMENT PRIMARY KEY,
   positionName VARCHAR(25) NOT NULL);

CREATE TABLE warehouse.employees(
   ID_employee MEDIUMINT UNSIGNED NOT NULL AUTO_INCREMENT PRIMARY KEY,
   lastName VARCHAR(25) NOT NULL,
            firstName VARCHAR (25) NOT NULL,
            login VARCHAR(128) NOT NULL,
            password VARCHAR(128) NOT NULL,
            phoneNumber INT(9) UNSIGNED NOT NULL,
            email VARCHAR(50),
            street VARCHAR(50) NOT NULL,
   city VARCHAR(50) NOT NULL,
            ID_position MEDIUMINT UNSIGNED,
   FOREIGN KEY(ID_position) REFERENCES warehouse.positions(ID_position));

CREATE TABLE warehouse.providers(
   ID_provider MEDIUMINT UNSIGNED NOT NULL AUTO_INCREMENT PRIMARY KEY,
   providerName VARCHAR(100) NOT NULL,
            phoneNumber INT(9) UNSIGNED NOT NULL,
            email VARCHAR(50),
            street VARCHAR(50) NOT NULL,
   city VARCHAR(50) NOT NULL);

CREATE TABLE warehouse.packs(
   ID_pack MEDIUMINT UNSIGNED NOT NULL AUTO_INCREMENT PRIMARY KEY,
   capacity INT(5) NOT NULL,
   material VARCHAR(25) NOT NULL);

CREATE TABLE warehouse.consolidatedPacks(
   ID_consolidatedPack MEDIUMINT UNSIGNED NOT NULL AUTO_INCREMENT PRIMARY KEY,
   quantity MEDIUMINT NOT NULL,
```

```
    packType VARCHAR(50) NOT NULL,
            dimensions VARCHAR(50));

CREATE TABLE warehouse.orderStates(
    ID_orderState MEDIUMINT UNSIGNED NOT NULL AUTO_INCREMENT PRIMARY KEY,
    stateName VARCHAR(50) NOT NULL);

CREATE TABLE warehouse.drinks(
    ID_drink MEDIUMINT UNSIGNED NOT NULL AUTO_INCREMENT PRIMARY KEY,
    drinkName VARCHAR(50) NOT NULL,
    providersPrice DECIMAL(6,2) NOT NULL,
            withSugar BOOLEAN NOT NULL,
            withAlcohol BOOLEAN NOT NULL,
    alcoholDose DECIMAL(4,2) NOT NULL,
    fizzy BOOLEAN NOT NULL,
            warehouseSector VARCHAR(25),
            stockAmount INT UNSIGNED NOT NULL,
    ID_pack MEDIUMINT UNSIGNED NOT NULL,
    ID_consolidatedPack MEDIUMINT UNSIGNED NOT NULL,
            FOREIGN KEY(ID_pack) REFERENCEs warehouse.packs(ID_pack),
    FOREIGN KEY(ID_consolidatedPack) REFERENCES
warehouse.consolidatedPacks(ID_consolidatedPack));

CREATE TABLE warehouse.clientOrders(
    ID_clientOrder MEDIUMINT UNSIGNED NOT NULL AUTO_INCREMENT PRIMARY KEY,
    fillingDate DATE NOT NULL,
    ID_orderState MEDIUMINT UNSIGNED NOT NULL,
    ID_client MEDIUMINT UNSIGNED NOT NULL,
    ID_employee MEDIUMINT UNSIGNED NOT NULL);

alter table warehouse.clientOrders add FOREIGN key (ID_orderState) REFERENCES
warehouse.orderStates(ID_orderState);
alter table warehouse.clientOrders add FOREIGN key (ID_client) REFERENCES warehouse.clients(ID_client);
alter table warehouse.clientOrders add FOREIGN key (ID_employee) REFERENCES
warehouse.employees(ID_employee);

CREATE TABLE warehouse.clientOrderedDrinks(
    ID_clientOrderedDrink MEDIUMINT UNSIGNED NOT NULL AUTO_INCREMENT PRIMARY KEY,
    ID_clientOrder MEDIUMINT UNSIGNED NOT NULL,
    ID_drink MEDIUMINT UNSIGNED NOT NULL,
    amount MEDIUMINT UNSIGNED NOT NULL,
    totalValue DECIMAL(10,2) NOT NULL);

alter table warehouse.clientOrderedDrinks add FOREIGN key (ID_clientOrder) REFERENCES
warehouse.clientOrders(ID_clientOrder);
alter table warehouse.clientOrderedDrinks add FOREIGN key (ID_drink) REFERENCES
warehouse.drinks(ID_drink);

CREATE TABLE warehouse.warehouseOrders(
    ID_warehouseOrder MEDIUMINT UNSIGNED NOT NULL AUTO_INCREMENT PRIMARY KEY,
    fillingDate DATE NOT NULL,
```

```
   automaticOrder BOOLEAN NOT NULL,
   ID_orderState MEDIUMINT UNSIGNED NOT NULL,
   ID_provider MEDIUMINT UNSIGNED NOT NULL,
   ID_employee MEDIUMINT UNSIGNED NOT NULL);

alter table warehouse.warehouseOrders add FOREIGN key (ID_orderState) REFERENCES
warehouse.orderStates(ID_orderState);
alter table warehouse.warehouseOrders add FOREIGN key (ID_provider) REFERENCES
warehouse.providers(ID_provider);
alter table warehouse.warehouseOrders add FOREIGN key (ID_employee) REFERENCES
warehouse.employees(ID_employee);

CREATE TABLE warehouse.warehouseOrderedProducts(
   ID_warehouseOrderedProduct MEDIUMINT UNSIGNED NOT NULL AUTO_INCREMENT PRIMARY
KEY,
   ID_warehouseOrder MEDIUMINT UNSIGNED NOT NULL,
            ID_drink MEDIUMINT UNSIGNED NOT NULL,
            amount MEDIUMINT UNSIGNED NOT NULL,
            totalValue DECIMAL(10,2) NOT NULL);

alter table warehouse.warehouseOrderedProducts add FOREIGN key (ID_warehouseOrder) REFERENCES
warehouse.warehouseOrders(ID_warehouseOrder);
alter table warehouse.warehouseOrderedProducts add FOREIGN key (ID_drink) REFERENCES
warehouse.drinks(ID_drink);


ALTER TABLE warehouse.clients ADD INDEX (clientName);
ALTER TABLE warehouse.drinks ADD INDEX (drinkName);


INSERT INTO `warehouse`.`providers` (`ID_provider`, `providerName`, `phoneNumber`, `email`, `street`,
`city`) VALUES (NULL, 'Dostawca numer 1', '111222333', NULL, 'Nowa 1', 'Nowgród 11-111');
INSERT INTO `warehouse`.`providers` (`ID_provider`, `providerName`, `phoneNumber`, `email`, `street`,
`city`) VALUES (NULL, 'Dostawca numer 2', '123456789', NULL, 'Druga 2', 'Drugogród 22-222');


INSERT INTO `warehouse`.`clients` (`ID_client`, `clientName`, `nip`, `phoneNumber`, `email`, `discount`,
`street`, `city`) VALUES (NULL, 'Kowalski Adam', '2455645874', '123123123', NULL, '0', 'Ulicowa 10',
'Miasto 66-666');
INSERT INTO `warehouse`.`clients` (`ID_client`, `clientName`, `nip`, `phoneNumber`, `email`, `discount`,
`street`, `city`) VALUES (NULL, 'Kowalska Agta', '9877899878', '555666444', NULL, '10', 'Lipna 2', 'Lipki
22-200');
INSERT INTO `warehouse`.`clients` (`ID_client`, `clientName`, `nip`, `phoneNumber`, `email`, `discount`,
`street`, `city`) VALUES (NULL, 'Cebulski Janusz', '7777777777', '741852963', NULL, '23', 'Długa 23',
'Starogród 12-657');


INSERT INTO `warehouse`.`packs` (`ID_pack`, `capacity`, `material`) VALUES (NULL, '500', 'karton');
INSERT INTO `warehouse`.`packs` (`ID_pack`, `capacity`, `material`) VALUES (NULL, '1500', 'plastik');
INSERT INTO `warehouse`.`packs` (`ID_pack`, `capacity`, `material`) VALUES (NULL, '2000', 'plastik');
```

```
INSERT INTO `warehouse`.`consolidatedPacks` (`ID_consolidatedPack`, `quantity`, `packType`, `dimensions`) VALUES (NULL, '250', 'zgrzewka', '1200x1000x1600');
INSERT INTO `warehouse`.`consolidatedPacks` (`ID_consolidatedPack`, `quantity`, `packType`, `dimensions`) VALUES (NULL, '400', 'karton', '1600x1600x1600');
INSERT INTO `warehouse`.`consolidatedPacks` (`ID_consolidatedPack`, `quantity`, `packType`, `dimensions`) VALUES (NULL, '100', 'zgrzewka', '800x800x1000');

INSERT INTO `warehouse`.`drinks` (`ID_drink`, `drinkName`, `providersPrice`, `withSugar`, `withAlcohol`, `alcoholDose`, `fizzy`, `ID_pack`, `ID_consolidatedPack`, `warehouseSector`, `stockAmount`) VALUES (NULL, 'Hortex jabłko mięta', '2,50', '0', '0', '0', '0', '1', '2', 'A2', '3200');
INSERT INTO `warehouse`.`drinks` (`ID_drink`, `drinkName`, `providersPrice`, `withSugar`, `withAlcohol`, `alcoholDose`, `fizzy`, `ID_pack`, `ID_consolidatedPack`, `warehouseSector`, `stockAmount`) VALUES (NULL, 'Polaris woda niegazowana', '0.99', '0', '0', '0', '0', '2', '1', 'C1', '5500');
INSERT INTO `warehouse`.`drinks` (`ID_drink`, `drinkName`, `providersPrice`, `withSugar`, `withAlcohol`, `alcoholDose`, `fizzy`, `ID_pack`, `ID_consolidatedPack`, `warehouseSector`, `stockAmount`) VALUES (NULL, 'Coca Cola', '3.90', '1', '0', '0', '1', '3', '3', 'D5', '10000');

INSERT INTO `warehouse`.`positions` (`ID_position`, `positionName`) VALUES (NULL, 'Administrator');
INSERT INTO `warehouse`.`positions` (`ID_position`, `positionName`) VALUES (NULL, 'Sprzedawca');
INSERT INTO `warehouse`.`positions` (`ID_position`, `positionName`) VALUES (NULL, 'Magazynier');

INSERT INTO `warehouse`.`employees` (`ID_employee`, `lastName`, `firstName`, `login`, `password`, `phoneNumber`, `email`, `street`, `city`, `ID_position`) VALUES (NULL, 'Nowak', 'Jan', 'c7ad44cbad762a5da0a452f9e854fdc1e0e7a52a38015f23f3eab1d80b931dd472634dfac71cd34ebc35d16ab7fb8a90c81f975113d6c7538dc69dd8de9077ec', 'c7ad44cbad762a5da0a452f9e854fdc1e0e7a52a38015f23f3eab1d80b931dd472634dfac71cd34ebc35d16ab7fb8a90c81f975113d6c7538dc69dd8de9077ec', '454565464', NULL, 'Menadżerska 5', 'Warszawa 30-300', '1');
INSERT INTO `warehouse`.`employees` (`ID_employee`, `lastName`, `firstName`, `login`, `password`, `phoneNumber`, `email`, `street`, `city`, `ID_position`) VALUES (NULL, 'Potocki', 'Adam', '5b42ee3081620cfc8da882a2ed4985a3c5cd56415da6ba8a3e644504030296742d7905c828bffd33c597cd54f2efd4e7984cf5bb6fe4a4f409a6d876e2d464a4', 'd44bd4b6bc03fdf28cda23877ee25e7a829ae8ece559965b20ada21f5457578c720eea1688efc282848b77cc1056b309fe2cf5006ccb8016c47a1d5702690c0d', '112223331', NULL, 'Bazarska 3', 'Sopot 20-300', '2');
INSERT INTO `warehouse`.`employees` (`ID_employee`, `lastName`, `firstName`, `login`, `password`, `phoneNumber`, `email`, `street`, `city`, `ID_position`) VALUES (NULL, 'Chudomski', 'Zbigniew', '8958123ae707ebca7eaaa0fb051801cb86ba13640e2bc5315c7e1dd9f846e29412148b6351427883f21551d394cbaf3ff2ec613b741e73eab1848fbf3d8c54fd', 'de43e2d46ff093ede0c14729814d07760bc1b5da59c76c39298d13ff8591228f293bc5003fca688e76d45010b5206aa9b72b84ef3a54e9b91ca57abec44abc03', '998889987', NULL, 'Porządkowa 17/1', 'Poznań 30-100', '3');

INSERT INTO `warehouse`.`orderStates` (`ID_orderState`, `stateName`) VALUES (NULL, 'Złożono');
INSERT INTO `warehouse`.`orderStates` (`ID_orderState`, `stateName`) VALUES (NULL, 'W realizacji');
INSERT INTO `warehouse`.`orderStates` (`ID_orderState`, `stateName`) VALUES (NULL, 'Zrealizowano');
INSERT INTO `warehouse`.`orderStates` (`ID_orderState`, `stateName`) VALUES (NULL, 'Anulowano');



DELIMITER //
CREATE PROCEDURE AddClient(
    IN clientName VARCHAR(100),
    IN nip INT(10) UNSIGNED,
```

```
   IN phoneNumber int(9) UNSIGNED,
   IN email VARCHAR(50),
   IN discount INT(3),
   IN street VARCHAR(50),
   IN city VARCHAR(50))
BEGIN
INSERT INTO `warehouse`.`clients` (`ID_client`, `clientName`, `nip`, `phoneNumber`, `email`, `discount`,
`street`, `city`) VALUES (NULL, clientName, nip, phoneNumber, email, discount, street, city);
END //
DELIMITER ;
CALL AddClient('testowanazwa', 6666666666, 666666666, NULL, 0, 'testowaulica 6', 'testowemiasto 66-666');

DELIMITER //
CREATE PROCEDURE AddWorker(
   IN lastName VARCHAR(25),
            IN firstName VARCHAR (25),
            IN login VARCHAR(128),
            IN password VARCHAR(128),
            IN phoneNumber INT(9) UNSIGNED,
            IN email VARCHAR(50),
            IN street VARCHAR(50),
   IN city VARCHAR(50),
            IN ID_position MEDIUMINT UNSIGNED)
BEGIN
INSERT INTO `warehouse`.`employees` (`ID_employee`, `lastName`, `firstName`, `login`, `password`,
`phoneNumber`, `email`, `street`, `city`, `ID_position`) VALUES (NULL, lastName, firstName, login,
password, phoneNumber, email, street, city, ID_position);
END //
DELIMITER ;

DELIMITER //
CREATE PROCEDURE AddProvider(
   IN providerName VARCHAR(100),
            IN phoneNumber INT(9) UNSIGNED,
            IN email VARCHAR(50),
            IN street VARCHAR(50),
   IN city VARCHAR(50))
BEGIN
INSERT INTO `warehouse`.`providers` (`ID_provider`, `providerName`, `phoneNumber`, `email`, `street`,
`city`) VALUES (NULL, providerName, phoneNumber, email, street, city);
END //
DELIMITER ;

DELIMITER //
CREATE PROCEDURE AddDrink(
   IN drinkName VARCHAR(50),
   IN providersPrice DECIMAL(6,2),
         IN withSugar BOOLEAN,
         IN withAlcohol BOOLEAN,
   IN alcoholDose DECIMAL(4,2),
   IN fizzy BOOLEAN,
```

```
        IN warehouseSector VARCHAR(25),
        IN stockAmount INT UNSIGNED,
    IN ID_pack MEDIUMINT UNSIGNED,
    IN ID_consolidatedPack MEDIUMINT UNSIGNED)
BEGIN
INSERT INTO `warehouse`.`drinks` (`ID_drink`, `drinkName`, `providersPrice`, `withSugar`, `withAlcohol`,
`alcoholDose`, `fizzy`, `ID_pack`, `ID_consolidatedPack`, `warehouseSector`, `stockAmount`) VALUES
(NULL, drinkName, providersPrice, withSugar, withAlcohol, alcoholDose, fizzy, ID_pack,
ID_consolidatedPack, warehouseSector, stockAmount);
END //
DELIMITER ;

DELIMITER //
CREATE PROCEDURE MakeClientOrder(
    IN fillingDate DATE,
    IN ID_orderState MEDIUMINT UNSIGNED,
    IN ID_client MEDIUMINT UNSIGNED,
    IN ID_employee MEDIUMINT UNSIGNED)
BEGIN
INSERT INTO `warehouse`.`clientOrders` (`ID_clientOrder`, `fillingDate`, `ID_orderState`, `ID_client`,
`ID_employee`) VALUES (NULL, fillingDate, ID_orderState, ID_client, ID_employee);
END //
DELIMITER ;

DELIMITER //
CREATE PROCEDURE MakeWarehouseOrder(
    IN fillingDate DATE,
    IN automaticOrder BOOLEAN,
    IN ID_orderState MEDIUMINT UNSIGNED,
    IN ID_provider MEDIUMINT UNSIGNED,
    IN ID_employee MEDIUMINT UNSIGNED)
BEGIN
INSERT INTO `warehouse`.`warehouseOrders` (`ID_warehouseOrder`, `fillingDate`, `automaticOrder`,
`ID_orderState`, `ID_provider`, `ID_employee`) VALUES (NULL, fillingDate, automaticOrder, ID_orderState,
ID_provider, ID_employee);
END //
DELIMITER ;

DELIMITER //
CREATE PROCEDURE AddOrderedWarehouseProduct(
                IN ID_warehouseOrder MEDIUMINT UNSIGNED,
                IN drinkID MEDIUMINT UNSIGNED,
                IN quantity MEDIUMINT UNSIGNED)
BEGIN
DECLARE totalValue DECIMAL(10,2);
SET totalValue = (SELECT providersPrice FROM warehouse.drinks WHERE ID_drink = drinkID)*quantity;
INSERT INTO `warehouse`.`warehouseOrderedProducts` (`ID_warehouseOrderedProduct`,
`ID_warehouseOrder`, `ID_drink`, `amount`, `totalValue`) VALUES (NULL, ID_warehouseOrder, drinkID,
quantity, totalValue);
END //
DELIMITER ;
```

```
DELIMITER //
CREATE PROCEDURE AddOrderedClientProduct(
   IN ID_clientOrder MEDIUMINT UNSIGNED,
   IN drinkID MEDIUMINT UNSIGNED,
   IN quantity MEDIUMINT UNSIGNED)
BEGIN
DECLARE totalValue DECIMAL(10,2);
SET totalValue = (SELECT providersPrice FROM warehouse.drinks WHERE ID_drink =
drinkID)*quantity*1.23;
INSERT INTO `warehouse`.`clientOrderedDrinks` (`ID_clientOrderedDrink`, `ID_clientOrder`, `ID_drink`,
`amount`, `totalValue`) VALUES (NULL, ID_clientOrder, drinkID, quantity, totalValue);
END //
DELIMITER ;


DELIMITER //
CREATE PROCEDURE DeleteDrink(IN id MEDIUMINT UNSIGNED)
BEGIN
DELETE FROM warehouse.drinks WHERE ID_drink = id;
END //
DELIMITER ;

DELIMITER //
CREATE PROCEDURE CancelClientOrder(IN id MEDIUMINT UNSIGNED)
BEGIN
DELETE FROM warehouse.clientOrderedDrinks WHERE ID_clientOrder = id;
UPDATE warehouse.clientOrders SET ID_orderState = 4 WHERE ID_clientOrder = id;
END //
DELIMITER ;

DELIMITER //
CREATE PROCEDURE CancelWarehouseOrder(IN id MEDIUMINT UNSIGNED)
BEGIN
DELETE FROM warehouse.warehouseOrderedProducts WHERE ID_warehouseOrder = id;
UPDATE warehouse.warehouseOrders SET ID_orderState = 4 WHERE ID_warehouseOrder = id;
END //
DELIMITER ;



DELIMITER //
CREATE PROCEDURE ChangeProductQuantity(IN productID MEDIUMINT UNSIGNED, IN newQuantity
INT UNSIGNED)
BEGIN
UPDATE warehouse.drinks SET stockAmount = newQuantity WHERE ID_drink = productID;
END //
DELIMITER ;

DELIMITER //
CREATE PROCEDURE EditProvider(
```

```
        IN PARAMproviderID MEDIUMINT UNSIGNED,
        IN PARAMproviderName VARCHAR(100),
        IN PARAMphoneNumber INT(9) UNSIGNED,
        IN PARAMemail VARCHAR(50),
        IN PARAMstreet VARCHAR(50),
   IN PARAMcity VARCHAR(50))
BEGIN
UPDATE warehouse.providers SET providerName = PARAMproviderName, phoneNumber =
PARAMphoneNumber, email = PARAMemail, street = PARAMstreet, city = PARAMcity WHERE
ID_provider = PARAMproviderID;
END //
DELIMITER ;

DELIMITER //
CREATE PROCEDURE EditClient(
        IN PARAMclientID MEDIUMINT UNSIGNED,
        IN PARAMclientName VARCHAR(100),
   IN PARAMnip INT(10) UNSIGNED,
   IN PARAMphoneNumber int(9) UNSIGNED,
   IN PARAMemail VARCHAR(50),
   IN PARAMdiscount INT(3),
   IN PARAMstreet VARCHAR(50),
   IN PARAMcity VARCHAR(50))
BEGIN
UPDATE warehouse.clients SET clientName = PARAMclientName, nip = PARAMnip, phoneNumber =
PARAMphoneNumber, email = PARAMemail, discount = PARAMdiscount, street = PARAMstreet, city =
PARAMcity WHERE ID_client = PARAMclientID;
END //
DELIMITER ;

DELIMITER //
CREATE PROCEDURE EditWorker(
        IN PARAMemployeeID MEDIUMINT UNSIGNED,
        IN PARAMlastName VARCHAR(25),
        IN PARAMfirstName VARCHAR (25),
        IN PARAMlogin VARCHAR(128),
        IN PARAMpassword VARCHAR(128),
        IN PARAMphoneNumber INT(9) UNSIGNED,
        IN PARAMemail VARCHAR(50),
        IN PARAMstreet VARCHAR(50),
        IN PARAMcity VARCHAR(50),
        IN PARAMID_position MEDIUMINT UNSIGNED)
BEGIN
UPDATE warehouse.employees SET lastName = PARAMlastName, firstName = PARAMfirstName, login =
PARAMlogin, password = PARAMpassword, phoneNumber = PARAMphoneNumber, email = PARAMemail,
street = PARAMstreet, city = PARAMcity, ID_position = PARAMID_position WHERE ID_employee =
PARAMemployeeID;
END //
DELIMITER ;

DELIMITER //
```

```
CREATE PROCEDURE EditOrderedWarehouseProduct(
        IN PARAMID_orderedWarehouseProduct MEDIUMINT UNSIGNED,
        IN PARAMID_warehouseOrder MEDIUMINT UNSIGNED,
        IN PARAMdrinkID MEDIUMINT UNSIGNED,
        IN PARAMquantity MEDIUMINT UNSIGNED)
BEGIN
DECLARE totalV DECIMAL(10,2);
SET totalV = (SELECT providersPrice FROM warehouse.drinks WHERE ID_drink =
PARAMdrinkID)*PARAMquantity;
UPDATE warehouse.warehouseOrderedProducts SET ID_warehouseOrder = PARAMID_warehouseOrder,
ID_drink = PARAMdrinkID, amount = PARAMquantity, totalValue = totalV WHERE
ID_warehouseOrderedProduct = PARAMID_orderedWarehouseProduct;
END //
DELIMITER ;

DELIMITER //
CREATE PROCEDURE EditOrderedClientProduct(
        IN PARAMID_orderedClientProduct MEDIUMINT UNSIGNED,
   IN PARAMID_clientOrder MEDIUMINT UNSIGNED,
   IN PARAMdrinkID MEDIUMINT UNSIGNED,
   IN PARAMquantity MEDIUMINT UNSIGNED)
BEGIN
DECLARE totalV DECIMAL(10,2);
SET totalV = (SELECT providersPrice FROM warehouse.drinks WHERE ID_drink =
PARAMdrinkID)*PARAMquantity*1.23;
UPDATE warehouse.clientOrderedDrinks SET ID_clientOrder = PARAMID_clientOrder, ID_drink =
PARAMdrinkID, amount = PARAMquantity, totalValue = totalV WHERE ID_clientOrderedDrink =
PARAMID_orderedClientProduct;
END //
DELIMITER ;

DELIMITER //
CREATE PROCEDURE EditDrink(
        IN PARAMID_drink MEDIUMINT UNSIGNED,
   IN PARAMdrinkName VARCHAR(50),
   IN PARAMprovidersPrice DECIMAL(6,2),
        IN PARAMwithSugar BOOLEAN,
        IN PARAMwithAlcohol BOOLEAN,
   IN PARAMalcoholDose DECIMAL(4,2),
   IN PARAMfizzy BOOLEAN,
        IN PARAMwarehouseSector VARCHAR(25),
        IN PARAMstockAmount INT UNSIGNED,
   IN PARAMID_pack MEDIUMINT UNSIGNED,
   IN PARAMID_consolidatedPack MEDIUMINT UNSIGNED)
BEGIN
UPDATE warehouse.drinks SET drinkName = PARAMdrinkName, providersPrice = PARAMprovidersPrice,
withSugar = PARAMwithSugar, withAlcohol = PARAMwithAlcohol, alcoholDose = PARAMalcoholDose,
fizzy = PARAMfizzy, ID_pack = PARAMID_pack, ID_consolidatedPack = PARAMID_consolidatedPack,
warehouseSector = PARAMwarehouseSector, stockAmount = PARAMstockAmount WHERE ID_drink =
PARAMID_drink;
END //
```

```
DELIMITER ;



DELIMITER //
CREATE PROCEDURE FindAllOrdersInProgress()
BEGIN
SELECT *
FROM(SELECT * FROM warehouse.clientOrders WHERE ID_orderState = 2) AS T1
NATURAL JOIN(SELECT * FROM warehouse.warehouseOrders WHERE ID_orderState = 2) AS T2;
END //
DELIMITER ;

DELIMITER //
CREATE PROCEDURE FindWarehouseOrderByDate(IN searchDate DATE)
BEGIN
SELECT * FROM warehouse.warehouseOrderedProducts WHERE searchData = fillingDate;
END //
DELIMITER ;

DELIMITER //
CREATE PROCEDURE FindWarehouseOrderByProvider(IN provider VARCHAR(100))
BEGIN
SELECT * FROM warehouse.warehouseOrderedProducts
        JOIN warehouse.providers ON ID_provider
  WHERE warehouse.providers.providerName LIKE provider;
END //
DELIMITER ;

DELIMITER //
CREATE PROCEDURE FindWarehouseOrderByWorker(IN name VARCHAR(25))
BEGIN
SELECT * FROM warehouse.warehouseOrderedProducts
        JOIN warehouse.employees ON ID_employee
  WHERE warehouse.employees.lastName LIKE name;
END //
DELIMITER ;

DELIMITER //
CREATE PROCEDURE FindDrink(IN name VARCHAR(50))
BEGIN
SELECT * FROM warehouse.warehouseOrderedProducts
  WHERE warehouse.drinks.drinkName LIKE name;
END //
DELIMITER ;

DELIMITER //
CREATE PROCEDURE FindClientOrder(IN name VARCHAR(100))
BEGIN
SELECT * FROM warehouse.clientOrders
        JOIN warehouse.clients ON ID_client
```

```
   WHERE warehouse.clients.clientName LIKE name AND warehouse.clientOrders.ID_orderState = 2;
END //
DELIMITER ;

DELIMITER //
CREATE PROCEDURE FindProvider(IN name VARCHAR(100))
BEGIN
SELECT * FROM warehouse.providers
   WHERE providerName LIKE name;
END //
DELIMITER ;

DELIMITER //
CREATE PROCEDURE FindClientByName(IN name VARCHAR(100))
BEGIN
SELECT * FROM warehouse.clients
   WHERE clientName LIKE name;
END //
DELIMITER ;

DELIMITER //
CREATE PROCEDURE FindClientByNIP(IN NIPnumber INT(10))
BEGIN
SELECT * FROM warehouse.clients
   WHERE nip = NIPnumber;
END //
DELIMITER ;

DELIMITER //
CREATE PROCEDURE FindWorkerByName(IN name VARCHAR(25))
BEGIN
SELECT * FROM warehouse.employees
   WHERE lastName LIKE name;
END //
DELIMITER ;
```