POLITECHNIKA WARSZAWSKA
WYDZIAŁ ELEKTRONIKI I TECHNIK INFORMACYJNYCH
TELEKOMUNIKACJA

BAZY DANYCH I BIG DATA

Hurtownia części elektronicznych

Autorzy:
Anh Quan Do 325 268
Agata Zatorska 325 338

Prowadzący: dr hab. inż Marcin Kowalczyk

Spis treści

1 Zakres i cel projektu

Celem projektu było zaprojektowanie oraz zaimplementowanie "relacyjnej bazy danych". Obejmowało to fazę konceptualną, w tym wyznaczenie modelu ER na poziomie konceptualnym, fazę projektowania logicznego, w której należało wyznaczyć schemat ER na poziomie logicznym, oraz fazę fizyczną, która polegała na implementacji fizycznej schematu ER.

W tym celu wykorzystaliśmy następujące programy:

- TOAD Data Modeler
- SQL Developer
- Oracle Database 19c

1.1 Opis założeń funkcjonalnych

Realizowana przez nas baza danych dotyczy hurtowni części elektronicznych. Przedsiębiorstwo to zajmuje się hurtowym handlem elektroniką.

Asortyment jest przechowywany w magazynach a do jego transportu wykorzystywane są dwa rodzaje pojazdów: ciężarówki do przewozu produktów na większą odległość oraz wózki widłowe wykorzystywane na halach.

Hurtownia zatrudnia pracowników na dwóch rodzajach stanowisk: magazyniera - pracującego na magazynie, który może prowadzić wózek widłowy (jeżeli ma odpowiednie uprawnienia) oraz kierowcę - obsługującego ciężarówki. W przypadku kierowcy przechowujemy także informacje o rodzaju posiadanych przez niego uprawnień do kierowania pojazdów.

Hurtownia ma w ofercie różnego rodzaju produkty z branży elektroniki, które mogą zostać zakupione. Ponieważ złożenie zamówienia wymaga podanie przez klienta danych personalnych, utrzymujemy również bazę klientów.

2 Definicja systemu

2.1 Perspektywy użytkowników

Wśród użytkowników, którzy mogliby mieć styczność oraz wykorzystywać naszą bazę danych możemy wyróżnić:

- **Główny zarządca Hurtowni Części Elektronicznych** - najwyższa pozycja i największy zakres dostępu do bazy danych. Ma prawo do wglądu i modyfikacji wszystkich danych w bazie.
- **Pracownik** - Ten użytkownik ma wgląd do danych związanych z działaniem Hurtowni: dane osobiste, kontaktowe oraz wynagrodzenie wszystkich **pracowników**; informacje dotyczące **magazynu** takie jak dostępność produktów oraz informacje dotyczące **pojazdów**. Może tylko i wyłącznie modyfikować dane **produktów** np. ich dostępność (tyczy się to tylko magazyniera).
- **Klient** - Ma dostęp do wglądu tylko ograniczonej liczby danych, takich jak informacje dotyczące produktów (części elektronicznych) lub czy może dany produkt kupić.

Definicja systemu zakłada również następujące funkcjonalności:

- Podgląd informacji o hurtowni
- Dodawanie/Modyfikowanie/Usuwanie informacji o hurtowni
- Podgląd informacji o pojazdach
- Dodawanie/Modyfikowanie/Usuwanie informacji o pojazdach
- Podgląd informacji o pojazdach jako ciężarówkach

- Dodawanie/Modyfikowanie/Usuwanie informacji o pojazdach jako ciężarówkach
- Podgląd informacji o pojazdach jako wózkach widłowych
- Dodawanie/Modyfikowanie/Usuwanie informacji o wózkach widłowych
- Podgląd informacji o magazynach
- Dodawanie/Modyfikowanie/Usuwanie informacji o magazynach
- Podgląd informacji o klientach
- Dodawanie/Modyfikowanie/Usuwanie informacji o klientach
- Podgląd informacji o pracownikach
- Dodawanie/Modyfikowanie/Usuwanie informacji o pracownikach
- Podgląd informacji o pracownikach jako magazynierach
- Dodawanie/Modyfikowanie/Usuwanie informacji o pracownikach jako magazynierach
- Podgląd informacji o pracownikach jako kierowcach
- Dodawanie/Modyfikowanie/Usuwanie informacji o kierowcach
- Podgląd informacji o produktach
- Dodawanie/Modyfikowanie/Usuwanie informacji o produktach
- Podgląd informacji o pojazdach obsługiwanych przez kierowców
- Dodawanie/Modyfikowanie/Usuwanie informacji o pojazdach obsługiwanych przez kierowców
- Podgląd informacji o magazynach w których pracują magazynierzy
- Dodawanie/Modyfikowanie/Usuwanie informacji o magazynach w których pracują magazynierzy
- Podgląd informacji o magazynach przechowujących produkty
- Dodawanie/Modyfikowanie/Usuwanie informacji o magazynach przechowujących produkty
- Podgląd informacji o produktach kupionych przez klientów
- Dodawanie/Modyfikowanie/Usuwanie informacji o produktach kupionych przez klientów

3 Model Konceptualny

3.1 Definicja zbiorów encji określonych w projekcie (decyzje projektowe)

W modelu konceptualnym wyróżniliśmy początkowo następujące encje:

- **Hurtownia** - główna encja, która zawiera wszystkie główne informacje dotyczące hurtowni części elektronicznych
- **Pojazd** - encja zawierająca informacje ogólne dotyczące pojazdów hurtowni
- **Ciezarowka** - encja uszczegóławiająca pojazd hurtowni jako ciężarówkę
- **Wozek_widlowy** - encja uszczegóławiająca pojazd hurtowni jako wózek widłowy
- **Magazyn** - encja zawierająca informacje ogólne dotyczące magazynów hurtowni

- **Klient** - encja zawierająca informacje ogólne dotyczące klientów hurtowni
- **Pracownik** - encja zawierająca informacje ogólne dotyczące pracowników hurtowni
- **Kierowca** - encja uszczegóławiająca pracownika hurtowni jako kierowcę
- **Magazynier** - encja uszczegóławiająca pracownika hurtowni jako magazyniera
- **Produkt** - encja zawierająca informacje ogólne dotyczące produktów hurtowni

3.2 Ustalenie związków między encjami i ich typów

Poniżej przedstawione są związki między encjami i ich typy oraz opisy:

Relacja	Krotność	Typ uczestnictwa	Opis
Hurtownia - Pracownik	Jeden do wielu	Hurtownia jest obowiązkowa dla pracownika, a pracownik jest opcjonalny dla hurtowni	W momencie założenia hurtowni może ona nie zatrudnić żadnego pracownika, a maksymalnie ma ich wielu. Pracownik żeby być zatrudnionym może pracować tylko w jednej hurtowni.
Hurtownia - Pojazd	Jeden do wielu	Hurtownia jest obowiązkowa dla pojazdu, a pojazd jest opcjonalny dla hurtowni	W momencie założenia hurtowni może ona nie mieć żadnego pojazdu, a maksymalnie ma ich wiele. Pojazd może należeć tylko do jednej hurtowni.
Hurtownia - Magazyn	Jeden do wielu	Hurtownia jest obowiązkowa dla magazynu, a magazyn jest opcjonalny dla hurtowni	W momencie założenia hurtowni może ona nie mieć żadnego magazynu, a maksymalnie ma ich wiele. Magazyn może należeć tylko do jednej hurtowni.
Hurtownia - Klient	Jeden do wielu	Hurtownia jest obowiązkowa dla klienta, a klient jest opcjonalny dla hurtowni	Hurtownia może nie mieć w pewnej chwili ani jednego klienta, ale może ich też mieć wiele. Dany klient może należeć tylko do jednej hurtowni.
Hurtownia - Produkt	Jeden do wielu	Hurtownia jest obowiązkowa dla produktu, a produkt jest opcjonalny dla hurtowni	Hurtownia może w momencie założenia nie mieć żadnego produktu, ale może ich też mieć wiele. Produkt może należeć tylko do jednej hurtowni jak już został dodany do bazy danych.
Produkt - Magazyn	Wiele do wielu	Produkt jest opcjonalny dla magazynu, a magazyn jest opcjonalna dla produktu	Magazyn może nie posiadać żadnego produktu (np. przed dostawą), ale może ich też mieć wiele. Produkt może nie należeć do żadnego magazynu, ale może też należeć do wielu magazynów.
Produkt - Klient	Wiele do wielu	Produkt jest opcjonalny dla klienta, a klient jest opcjonalna dla produktu	Produkt może być zakupiony przez klienta, ale może też nie być zakupiony przez nikogo. Klient może nie kupić żadnego produktu, ale może kupić też ich wiele.

Tabela 1: **Związki między encjami i ich typy oraz opisy**

3.3 Określenie atrybutów i ich dziedzin

Atrybut	Typ i dziedzina danych	Obowiązkowość	Opis
Nr_hurtowni	Integer	Obowiązkowy	Unikatowy identyfikator hurtowni.
Nazwa	VarChar(50)	Obowiązkowy	Nazwa hurtowni.
Adres	VarChar(400)	Obowiązkowy	Adres hurtowni. Pole segmentowe, które zawiera w sobie miasto, ulicę, numer budynku, numer lokalu i kod pocztowy.
Data_zalozenia	Date	Obowiązkowy	Data założenia hurtowni.
Wlasciciel	VarChar(50)	Obowiązkowy	Właściciel hurtowni. Pole segmentowe, które zawiera w sobie imię i nazwisko właściciela, datę urodzenia, PESEL, płeć, numer konta, e-mail i wynagrodzenie.

Tabela 2: **Atrybuty i dziedziny encji Hurtownia**

Atrybut	Typ i dziedzina danych	Obowiązkowość	Opis
Nr_pojazdu	Integer	Obowiązkowy	Unikatowy identyfikator pojazdu.
Marka	VarChar(50)	Obowiązkowy	Marka pojazdu. Pole segmentowe, które zawiera w sobie nazwę i datę założenia.
Model	VarChar(80)	Obowiązkowy	Model pojazdu. Pole segmentowe, które zawiera w sobie nazwę i rok produkcji.
Rok_produkcji	Char(4)	Obowiązkowy	Rok produkcji pojazdu.
Nr_rejestracyjny	VarChar(10)	Nieobowiązkowy	Numer rejestracyjny pojazdu.
VIN	Char(17)	Obowiązkowy	VIN (Vehicle Identification Number) pojazdu.
Pojemnosc_silnika	Integer	Obowiązkowy	Pojemnosc silnika pojazdu.
Rodzaj_paliwa	Rodzaj paliwaD ('LPG', 'b95', 'b98', 'diesel', 'elektryczny')	Obowiązkowy	Rodzaj paliwa pojazdu.

Tabela 3: **Atrybuty i dziedziny encji Pojazd**

Atrybut	Typ i dziedzina danych	Obowiązkowość	Opis
Pojemnosc	Integer	Obowiązkowy	Pojemność ładunkowa ciężarówki.
Typ_ciężarówki	Typ ciężarówkiD ('furgon', 'chlodnia', 'kontener', 'plandeka', 'siodlowowy')	Obowiązkowy	Typ ciężarówki.

Tabela 4: **Atrybuty i dziedziny encji Ciężarówka**

Atrybut	Typ i dziedzina danych	Obowiązkowość	Opis
Pojemnosc_udźwigu	Integer	Obowiązkowy	Pojemność udźwigu wózka widłowego.
Wysokość_unoszenia	Integer	Obowiązkowy	Wysokość unoszenia wózka widłowego.

Tabela 5: **Atrybuty i dziedziny encji Wózek widłowy**

Atrybut	Typ i dziedzina danych	Obowiązkowość	Opis
Nr_magazynu	Integer	Obowiązkowy	Unikatowy identyfikator magazynu.
Powierzchnia	Integer	Obowiązkowy	Powierzchnia magazynu.
Typ_magazynu	Typ magazynuD ('Chlodnia', 'Sezonowy', 'Cross-Docking', 'Centralny', 'Dystrybucyjny')	Obowiązkowy	Typ magazynu.
Nazwa	VarChar(30)	Obowiązkowy	Nazwa magazynu.
Pojemnosc	Integer	Obowiązkowy	Pojemność magazynu w paletach.
Adres	VarChar(400)	Obowiązkowy	Adres Magazynu. Pole segmentowe, które zawiera w sobie miasto, ulicę, numer budynku, numer lokalu i kod pocztowy.

Tabela 6: **Atrybuty i dziedziny encji Magazyn**

Atrybut	Typ i dziedzina danych	Obowiązkowość	Opis
Nr_klienta	Integer	Obowiązkowy	Unikatowy identyfikator klienta.
Imie	VarChar(30)	Obowiązkowy	Imię klienta.
Nazwisko	VarChar(50)	Obowiązkowy	Nazwisko klienta.
PESEL	Char(11)	Nieobowiązkowy	PESEL klienta.
Data_urodzenia	Date	Nieobowiązkowy	Data urodzenia klienta.
Pleć	PlecD ('K','M')	Nieobowiązkowy	Płeć klienta.
Email	VarChar(30)	Nieobowiązkowy	Adres e-mail klienta.
Nr Telefonu	VarChar(14)	Nieobowiązkowy	Numer telefonu klienta.
Adres	VarChar(400)	Obowiązkowy	Adres klienta. Pole segmentowe, które zawiera w sobie miasto, ulicę, numer budynku, numer lokalu i kod pocztowy.

Tabela 7: **Atrybuty i dziedziny encji Klient**

Atrybut	Typ i dziedzina danych	Obowiązkowość	Opis
Nr_pracownika	Integer	Obowiązkowy	Unikatowy identyfikator pracownika.
Imie	VarChar(30)	Obowiązkowy	Imię pracownika.
Nazwisko	VarChar(50)	Obowiązkowy	Nazwisko pracownika.
Data_urodzenia	Date	Obowiązkowy	Data urodzenia pracownika.
PESEL	Char(11)	Nieobowiązkowy	PESEL pracownika.
Adres	VarChar(400)	Obowiązkowy	Adres hurtowni. Pole segmentowe, które zawiera w sobie miasto, ulicę, numer budynku, numer lokalu i kod pocztowy.
Pleć	PlecD ('K','M')	Obowiązkowy	Płeć pracownika.
Stanowisko	VarChar(50)	Obowiązkowy	Stanowisko pracownika w hurtowni Pole segmentowe, które zawiera w sobie nazwę oraz opis.
Data_zatrudnienia	Date	Obowiązkowy	Data zatrudnienia pracownika.
Nr_konta	Char(34)	Nieobowiązkowy	Numer konta pracownika.
Email	VarChar(30)	Nieobowiązkowy	Adres e-mail pracownika.
Nr_Telefonu	VarChar(14)	Nieobowiązkowy	Numer telefonu pracownika.
Wynagrodzenie	Integer	Obowiązkowy	Wynagrodzenie pracownika.
Adres	VarChar(400)	Obowiązkowy	Adres Pracownika. Pole segmentowe, które zawiera w sobie miasto, ulicę, numer budynku, numer lokalu i kod pocztowy.

Tabela 8: **Atrybuty i dziedziny encji Pracownik**

Atrybut	Typ i dziedzina danych	Obowiązkowość	Opis
Nr_prawa_jazdy	Integer	Obowiązkowy	Numer prawa jazdy kierowcy.
Data_waznosci_prawa jazdy	Date	Obowiązkowy	Data waznosci prawa jazdy .

Tabela 9: **Atrybuty i dziedziny encji Kierowca**

Atrybut	Typ i dziedzina danych	Obowiązkowość	Opis
Szkolenie_BHP	Szkolenie BHPD ('T','N')	Obowiązkowy	Czy magazynier przeszedł szkolenie BHP.
Uprawnienia_wozek	Uprawnienia wozekD ('T','N')	Obowiązkowy	Czy magazynier ma uprawnienia na operowanie wózkiem widłowym.

Tabela 10: **Atrybuty i dziedziny encji Magazynier**

Atrybut	Typ i dziedzina danych	Obowiązkowość	Opis
Nr_produktu	Integer	Obowiązkowy	Unikatowy identyfikator produktu.
Nazwa	VarChar(50)	Obowiązkowy	Imię produktu.
Opis	VarChar(300)	Nieobowiązkowy	Opis produktu.
Ilosc_produktu	Integer	Obowiązkowy	Ilosc produktu.
Producent	VarChar(30)	Obowiązkowy	Producent produkujący produkt.
Cena	Integer	Obowiązkowy	Cena produktu.

Tabela 11: **Atrybuty i dziedziny encji Produkt**

3.4 Dodatkowe reguły integralnościowe (reguły biznesowe)

Dodatkowo, aby sprecyzować działanie modelu hurtowni dodaliśmy następujące reguły integralnościowe:

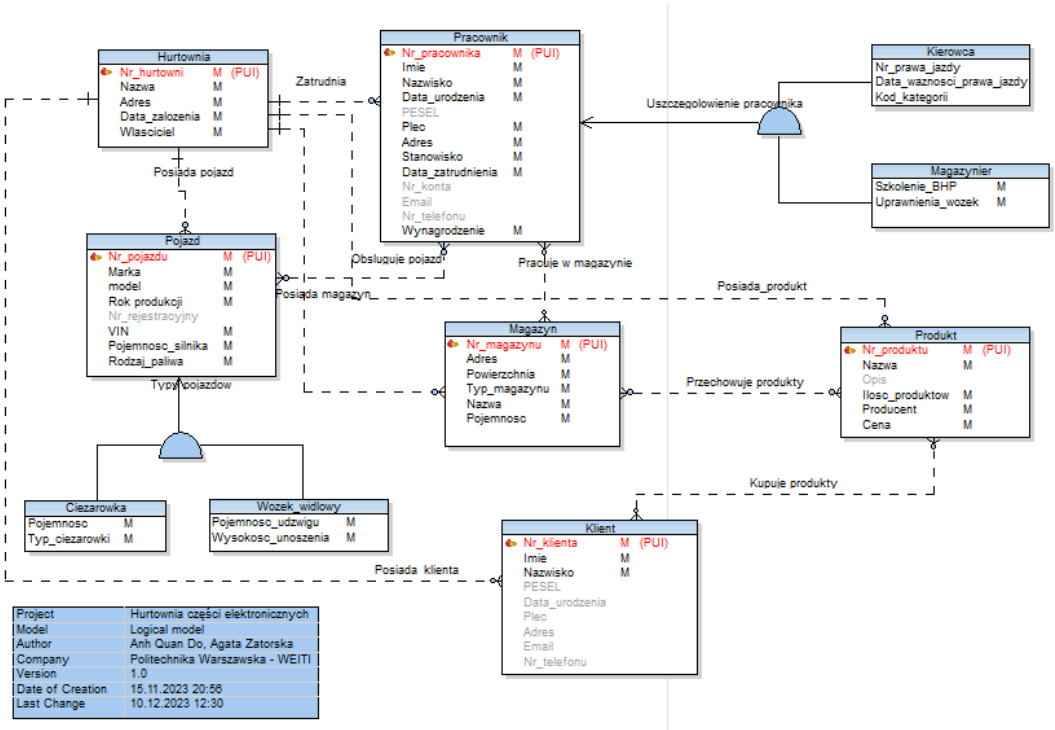
- Klient powinien zawsze mieć możliwość złożenia dużej ilości zamówień, a nie tylko pojedynczych zamówień na jeden produkt
- Klient powinien również mieć możliwość ponownego złożenia takiego samego zamówienia wiele razy.

3.5 Klucze kandydujące i główne (decyzje projektowe)

Nazwa encji	Klucz główny	Klucz kandydujący
Hurtownia	Nr_hurtowni	Nazwa
Pojazd	Nr_pojazdu	Model, VIN
Ciezarowka	Nr_pojazdu	Typ ciezarowki
Wozek_widlowy	Nr_pojazdu	-
Magazyn	Nr_magazynu	Typ magazynu, nazwa
Klient	Nr_klienta	Nazwisko, PESEL, Email, Nr_telefonu
Pracownik	Nr_pracownika	Nazwisko, PESEL, Email, Nr_telefonu
Kierowca	Nr_pracownika	Nr_prawa_jazdy
Magazynier	Nr_pracownika	-
Produkt	Nr_produktu	Nazwa

Tabela 12: **Klucze kandydujące i główne**

3.6 Schemat ER na poziomie konceptualnym

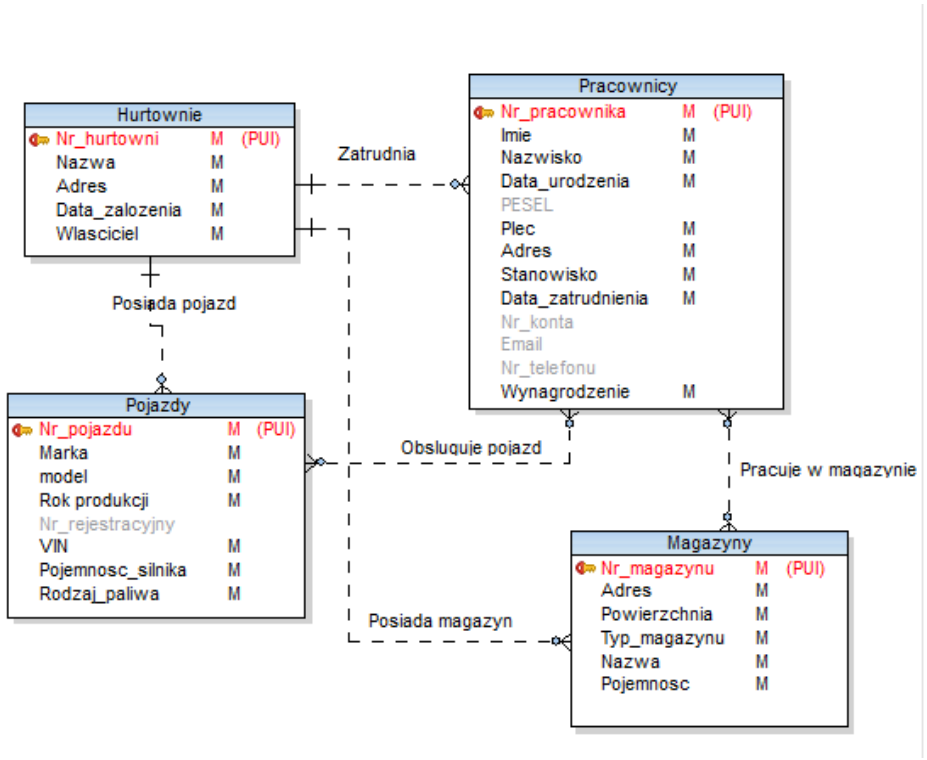


Rysunek 1: Schemat ER na poziomie konceptualnym

3.7 Problem pułapek szczelinowych i wachlarzowych – analiza i przykłady

Na obecnej fazie projektowania zaobserwowaliśmy dwa przypadki pułapek, jedną szczelinową i jedną wachlarzową.

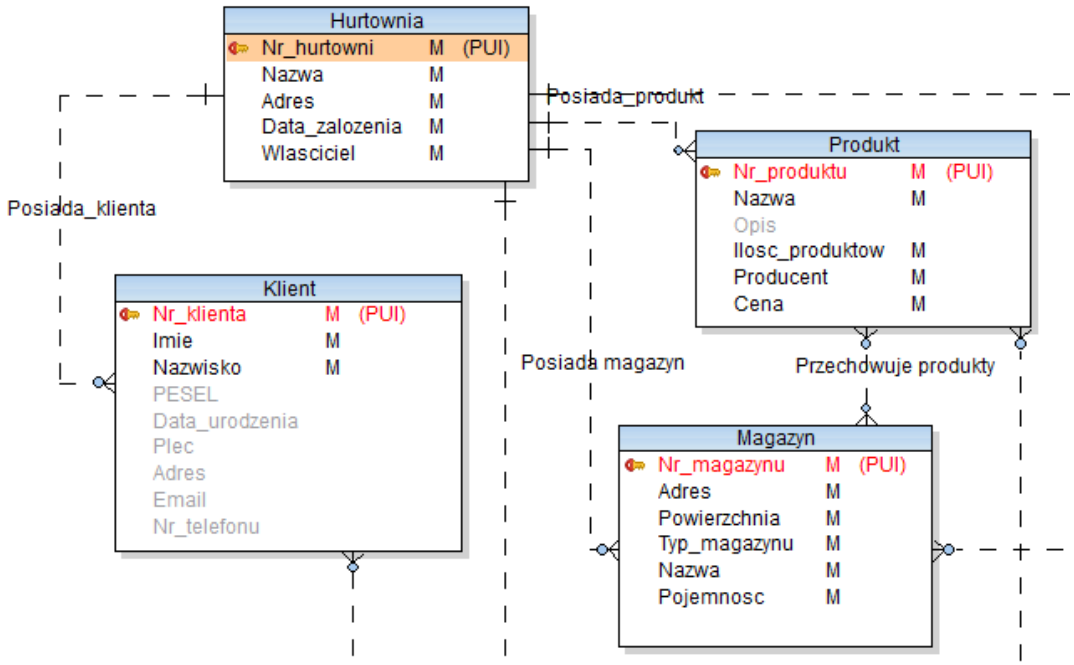
Pułapka szczelinowa zaistniałaby w przypadku, gdyby "Pracownik" był podłączony tylko i wyłącznie do encji "Pojazd" i "Magazyny" co oznaczałoby, że "Kierowca" mógłby być łączony z "Magazynem", a "Magazynier" z "Pojazdem", co jest nieprawdą. Z tego powodu połączyliśmy encję "Pracownik" z encją "Hurtownia" relacją "zatrudnia".



Rysunek 2: Pułapka szczelinowa

Pułapka wachlarzowa występuje dla takich samych encji, ale w innym przypadku. Dotyczy ona tego, że jeden pracownik może pracować w wielu magazynach lub obsługiwać wiele pojazdów i wyciągnięcie informacji o tym który pracownik pracuje w którym magazynie (analogicznie dla pojazdów) mogłoby doprowadzić do wachlarza możliwości. Aby uniknąć tego problemu połączyliśmy "Pracownika" z "Magazynem" relacją "pracuje w magazynie" i "Pracownika" z "Pojazdem" relacją "obsługuje pojazd".

Inne przykłady **pułapki szczelinowej** istnieją dla encji "Magazyn - Produkt" oraz "Produkt - Klient". W pierwszym przykładzie insynuowaliśmy, że każdy produkt powinien należeć do jakiegoś magazynu, a istnieją przypadki, w których produkty nie będą w magazynie (np. gdy zostały dopiero zamówione do magazynu lub gdy nie ma ich na stanie. Z tego powodu należało połączyć tabelę "Produkt" z "Hurtownią" relacją "Posiada_produkt, aby uniknąć tej pułapki. Analogicznie, przed poprawką można byłoby pomyśleć, że każdy klient musi mieć w w każdym momencie jakiś zamówiony produkt, a jest to nieprawda, gdyż klient mógł już mieć dostarczone produkty od jakiegoś czasu lub dopiero co się zarejestrować na stronie hurtowni, dlatego dodaliśmy połączenie "Posiada_klienta" pomiędzy "Hurtownią" a "Klientem".



Rysunek 3: Pułapka szczelinowa

4 Model Logiczny

4.1 Charakterystyka modelu relacyjnego

Po zakończeniu projektowania schematu ER na poziomie konceptualnym i sprawdzeniu poprawności przekonwertowaliśmy projekt na model logiczny korzystając z silnika bazy danych Oracle 19c. Po takiej konwersji otrzymaliśmy poniżej opisane zmiany:

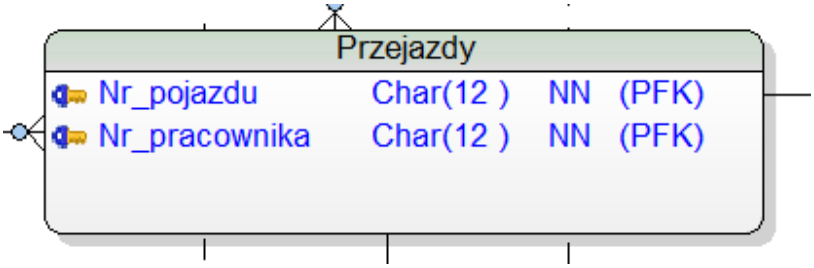
- Każda relacja "wielu do wielu" została zastąpiona nowo stworzoną tabelą, która połączona jest dwoma relacjami z oryginalnymi encjami. Relacje te z automatu wymuszają obowiązkowy typ uczestnictwa.
- Identyfikujące atrybuty każdej encji zostały zmienione na klucz główny tabeli. Dodatkowo, wszystkie encje uszczegóławiające zostały przekształcone w tabele o kluczu głównym encji nadrzędnej.
- Każdy związek "wielu do wielu" otrzymał klucz główny po jednej stronie związku oraz klucz obcy to jej drugiej stronie.
- Pomiędzy tabelami nadrzędnymi i uszczegóławiającymi zostały stworzone relacje "jeden do wielu" z obowiązkowym uczestnictwem rodzica i nieobowiązkowym uczestnictwem dziecka.
- Typy danych zostały przekonwertowane na te wykorzystywane w wybranym silniku bazy danych.

4.2 Usunięcie właściwości niekompatybilnych z modelem relacyjnym - przykłady

Na poziomie projektowania konceptualnego wszystkie encje (tablice w poziomie konceptualnym) pisaliśmy w liczbie pojedynczej. Aby odróżnić je od relacji (tablic w poziomie logicznym), przekształcamy nazwy wszystkich tablic na liczbę mnogą.

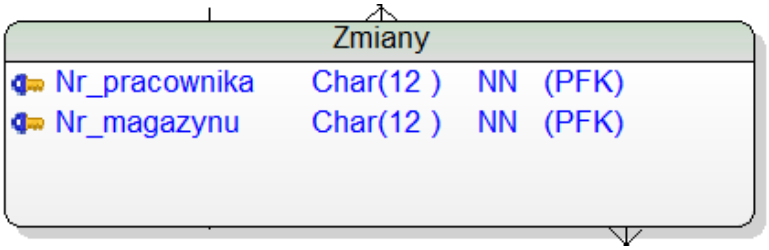
Konwersja automatycznie stworzyła nam tabele łączące w miejscach relacji "wiele do wielu", zatem musimy już tylko je dobrze nazwać dla przejrzystości. Takich tabel mieliśmy 4:

Tabelę pomiędzy "Pracownikami", a "Pojazdami" nazwaliśmy "Przejazdy". Zawiera ona w sobie informacje dotyczące numeru pojazdu, który jest obsługiwany przez pracownika (kierowcę) o odpowiednim numerze.



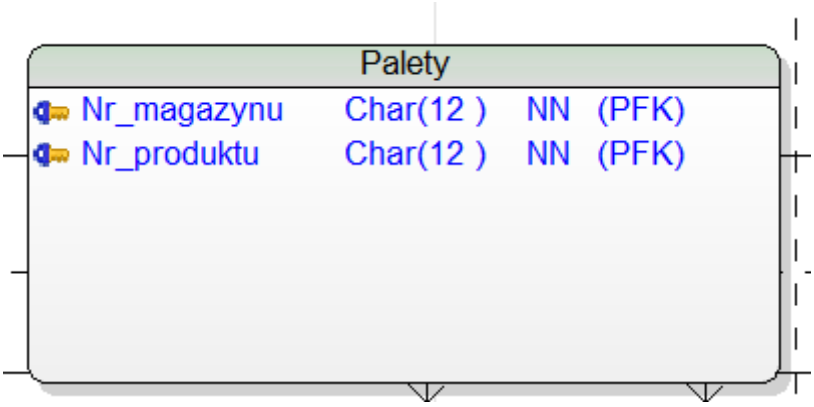
Rysunek 4: Tabela "Przejazdy"

Tabelę pomiędzy "Pracownikami", a "Magazynami" nazwaliśmy "Zmiany". Zawiera ona w sobie informacje dotyczące numeru magazynu, w który pracuje pracownik (magazynier) o odpowiednim numerze na swojej zmianie.



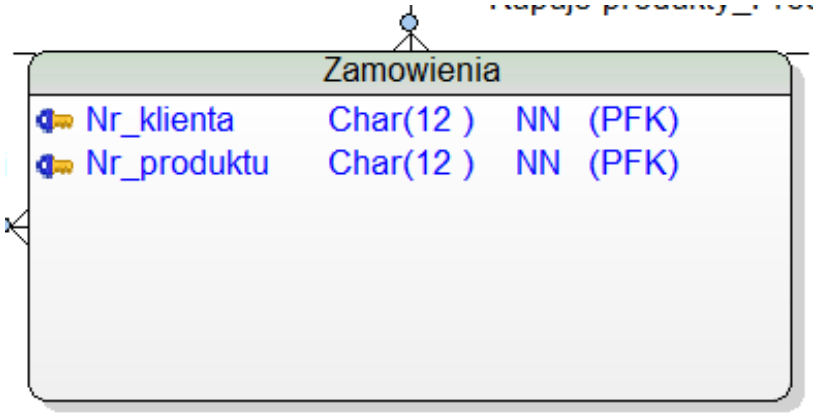
Rysunek 5: Tabela "Zmiany"

Tabelę pomiędzy "Produktami", a "Magazynami" nazwaliśmy "Palety". Zawiera ona w sobie informacje dotyczące numeru magazynu, w który znajdują się dane produkty o odpowiednim numerze.



Rysunek 6: Tabela "Palety"

Tabelę pomiędzy "Produktami", a "Klientami" nazwaliśmy "Zamówienia". Zawiera ona w sobie informacje dotyczące numeru produktów, które zamówił klient o danym numerze.



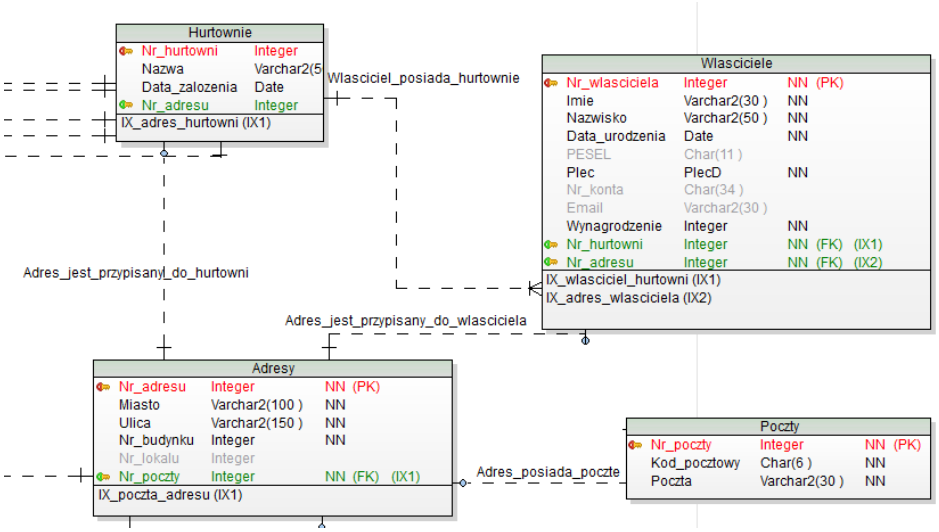
Rysunek 7: Tabela "Zamowienia"

4.3 Proces normalizacji – analiza i przykłady

Proces normalizacji w bazie danych robimy, aby wyeliminować powtarzające się dane i trzymać w jednym miejscu. Dzięki temu zyskujemy większe bezpieczeństwo oraz ułatwione i przyśpieszone aktualizowanie danych w przyszłości. Zmniejszamy również dzięki temu szansę na wszelakie anomalie i niespójności. Negatywem normalizacji jest jednak szansa na spowolniony odczyt danych ze względu na skomplikowany schemat danych.

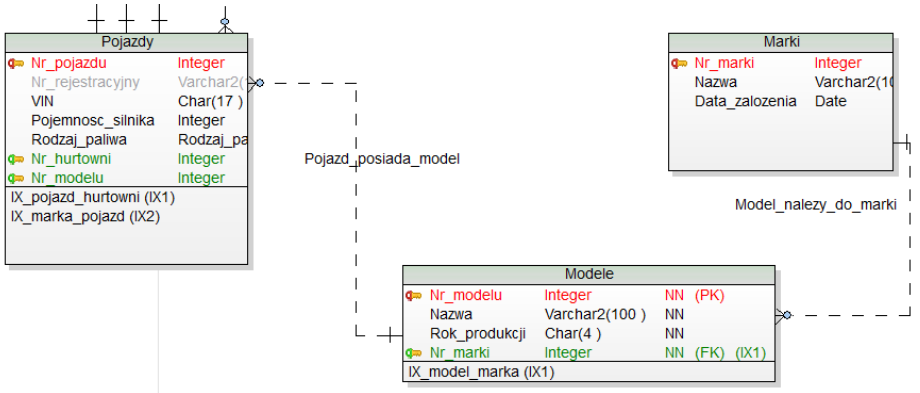
Normalizację zaczynamy od jej najniższego szczebla, czyli od pierwszej postaci normalnej (1NF). Jej wymogi są takie, aby każdy atrybut w danej tabeli był wartością atomową (czyli elementarną), co oznacza, że ma się nie rozkładać. Dodatkowo, w każdej relacji mają nie występować powtarzające się grupy. W naszym modelu znaleźliśmy poniższe przypadki, w których wymagana była zmiana:

- Atrybut adresy był polem segmentowym, ponieważ przechowywał w sobie więcej niż jeden typ wartości. Musieliśmy zatem wyodrębnić to pole i stworzyć nową relację "Adresy" zawierająca atrybuty: miasto, ulica, numer budynku i numer lokalu. Pole poczta zostało również wyciągnięte z relacji "Adresy" i zmienione w swoją oddzielną tabelę, a następnie połączyliśmy obie relacje ze sobą. Ostatecznie połączyliśmy relacje "Adresy" z wszystkimi innymi relacjami, gdzie poprzednio istniały atrybuty "adres" oraz usunęliśmy dany atrybut z tych relacji.
- Drugim atrybutem, który był polem segmentowy był atrybut "Właściciel". Zrobiliśmy zatem wszystko analogicznie tak jak dla "Adresu", czyli wyciągnęliśmy go z relacji "Hurtownia" i stworzyliśmy dla niego własną oddzielną tabelę z atrybutami: imię, nazwisko, data urodzenia, PESEL, płeć, numer konta, adres e-mail oraz wynagrodzenie. Musieliśmy również podłączyć teraz nowo powstałe relacje "Adresy" i "Właściciele".



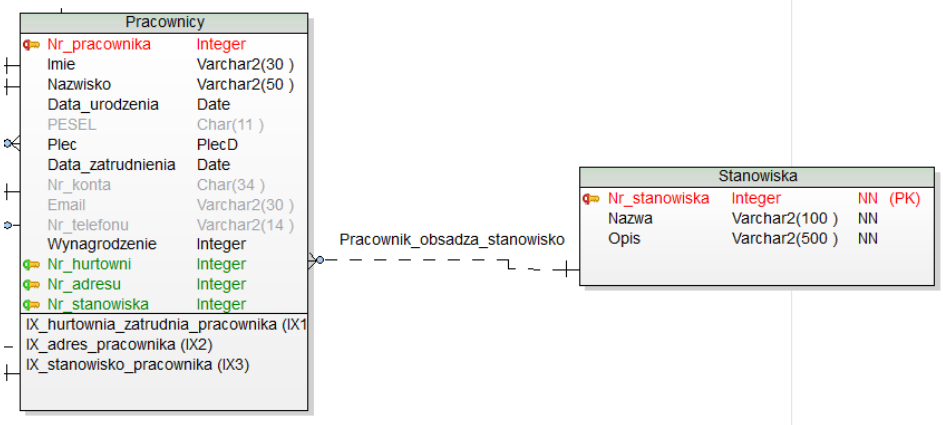
Rysunek 8: Normalizacja "Adresu", "Poczty" oraz "Właściciela"

- Kolejny atrybut, który był polem segmentowym, było pole "marka". Tak samo jak w poprzednich przykładach wyciągnęliśmy i stworzyliśmy nową tabelę, a w procesie zauważyliśmy, że pole "model" również jest polem segmentowym, zatem to też wyciągnęliśmy i stworzyliśmy nową relację, a potem wszystko ze sobą połączyliśmy.



Rysunek 9: Normalizacja "Modelu" i "Marki"

- Ostatnie pole segmentowe znaleźliśmy w atrybucie "stanowisko". Procedury z powyższych przykładów powtórzyliśmy i połączyliśmy ze sobą "Pracownika" i nowo powstałe "Stanowisko".

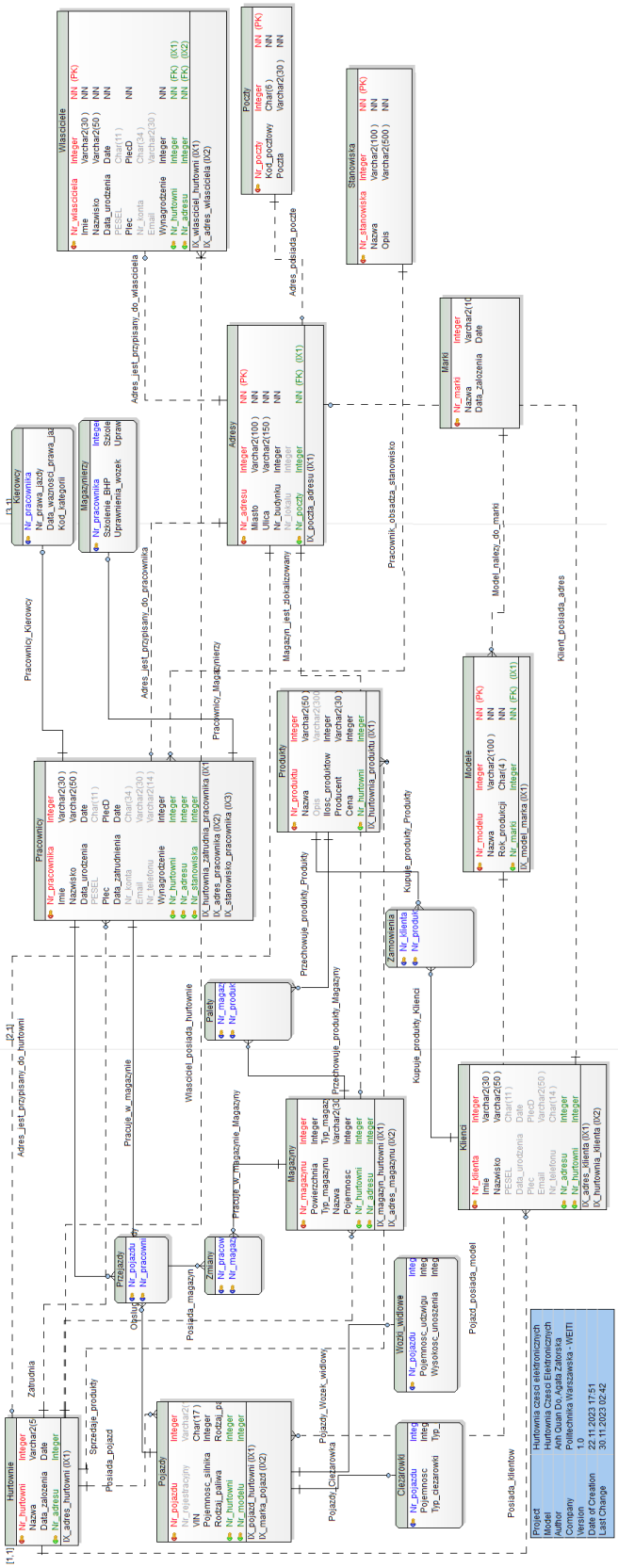


Rysunek 10: Normalizacja "Stanowiska"

Dla drugiej postaci normalnej (2NF) relacja musi być już w pierwszej postaci normalnej oraz każdy atrybut tej relacji nie wchodzący w skład żadnego klucza potencjalnego musi być w pełni funkcyjnie zależny od wszystkich kluczy potencjalnych tej relacji. Dodatkowo każdy atrybut tej relacji nie wchodzący w skład klucza musi zależeć od klucza, a nie tylko od jego części. Jako że każdy klucz główny w każdej relacji jest prosty (sami tworzyliśmy te "sztuczne" klucze o wartości Integer), to 2NF zostało spełnione.

Aby otrzymać trzecią postać normalną (3NF), relacja musi spełniać wymogi drugiej postaci normalnej oraz każdy atrybut relacji nie wchodzący w skład żadnego klucza potencjalnego nie może być przechodnio funkcyjnie zależny od żadnego klucza potencjalnego tej relacji. Ten warunek również został automatycznie spełniony.

4.4 Schemat ER na poziomie modelu logicznego



Rysunek 11: Schemat ER na poziomie modelu logicznego

4.5 Więzy integralności

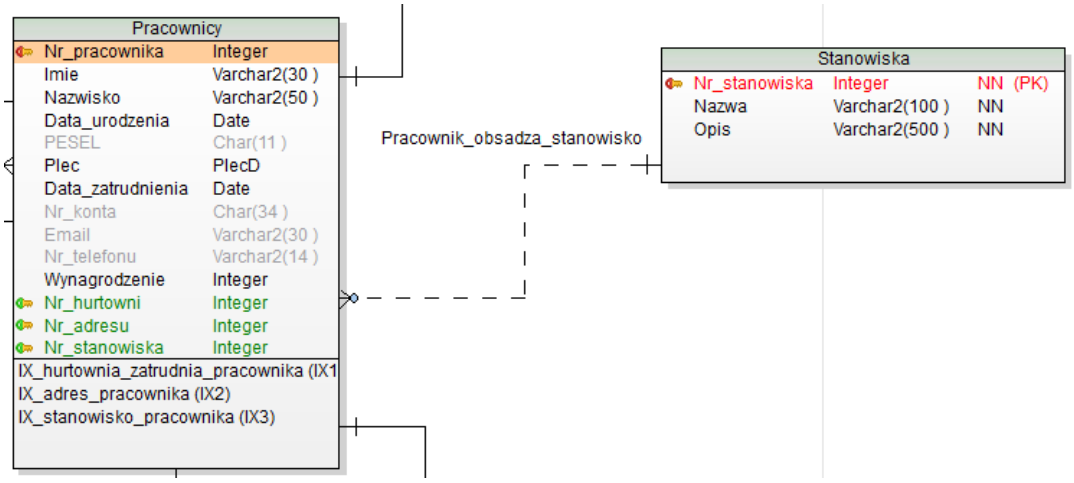
Więzy integralności to zagwarantowanie zgodności pomiędzy atrybutami, a typami, jakie te atrybuty mogą przyjąć. To oznacza, że jeśli jakiś atrybut został zdefiniowany jako specyficzny typ danych (np. Integer), to pole w bazie danych przechowujące dane związane z tym atrybutem nie może przyjąć innych typów danych (np. VarChar). Warunek ten jest spełniony w naszej bazie danych, ponieważ większość pól jest obowiązkowych o dokładnie określonym typie danych, które może przyjąć. W przypadku atrybutów, które nie są obowiązkowe, jedyna wartość jaką mogą przyjąć jest wartość NULL. Oprócz tego, każde pole kluczowe musi być unikatowe. To osiągnęliśmy już poprzez proces normalizacji.

4.6 Proces denormalizacji – analiza i przykłady

Proces denormalizacji to proces odwrotny do normalizacji. Polega on na przywróceniu stanu przed procesem normalizacji. Denormalizacja przyśpiesza dostęp do danych w bazie danych i dzięki temu zwiększa wydajność, jednakże robi to kosztem straty dokładności atrybutów oraz straty sprawności aktualizacji.

Aby pokazać i pozytywy, i negatywy denormalizacji posłużymy się dwoma przykładami.

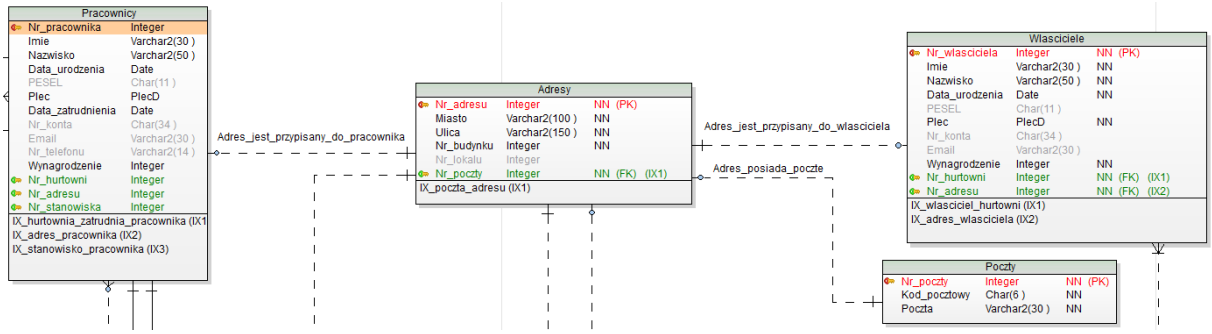
Pierwszy przykład, w którym pokażemy pozytywny wynik denormalizacji pokażemy na przykładzie korzystającym z nastepujących relacji:



Rysunek 12: Przykład do pozytywnej denormalizacji

W procesie denormalizacji na powyższym przykładzie usunęlibyśmy tabele "Stanowiska". Jak można zobaczyć, relacja stanowiska ma w sobie tylko dwa atrybuty, z czego jeden (opis) nie jest niezbędny, ani ważny. Usunięcie więc tej tabeli nie spowodowałoby dużą utratę dokładności. Dodatkowo "Stanowiska" połączone są tylko z jedną inną tabelą, a mianowicie z "Pracownikami". Wsadzenie z powrotem "Stanowisk" do "Pracowników" i pozostawienie tylko pola nazwa (jako VarChar) nie byłoby dla nas bardzo szkodliwe, a wręcz przyśpieszyłoby wyciąganie danych i dzięki temu zwiększyłaby się efektywność bazy danych.

W drugim przykładzie pokażemy jednak, że nieostrożna denormalizacja może prowadzić do dużych strat informacji - stracilibyśmy to, co chcieliśmy uzyskać w procesie normalizacji. Poniżej przedstawiono relacje, na których pokażemy nasz przykład:



Rysunek 13: Przykład do negatywnej denormalizacji

W tym przykładzie denormalizacja wiązałaby się z usunięciem relacji "Adres", ale na pierwszy rzut oka można już dojść do wniosku, że jest to raczej zły pomysł. W przeciwieństwie do poprzedniego przykładu, ta tabela ma w sobie o wiele więcej atrybutów, czyli redukcja tej tabeli oznaczałaby stratę wielu istotnych danych. Samo w sobie jest to już złe, ale dodatkowo "Adresy" połączone są z dwoma widocznymi na powyższym zdjęciu tabelami (oraz z trzema innymi niewidocznymi na zdjęciu), co sprawia, że strata informacji jest jeszcze większa. Z tego powodu, w tym przypadku lepiej byłoby zostawić już formę po normalizacji i nie przystępować do denormalizacji.

Wyciagając już z tego wnioski, denormalizacja w niektórych przypadkach jest wskazana, jednakże trzeba się zastanowić, czy uzyskana poprawa efektywności bazy danych jest warta straty informacji i trudniejszej aktualizacji w późniejszym etapie. W przypadku gdy relacja ma mało atrybutów i ma mało połączeń, denormalizacja może być pozytywna, natomiast gdy tych atrybutów jest dużo i ma dużo połączeń denormalizacja może mieć bardziej negatywne skutki.

5 Faza fizyczna

5.1 Projekt transakcji i weryfikacja ich wykonalności

Transakcja	Weryfikacja wykonalności	Potrzebne dane
Podgląd informacji o hurtowni	Wykonalne	Hurtownie, Adresy
Dodawanie/Modyfikowanie/Usuwanie informacji o hurtowni	Wykonalne	Hurtownie, Adresy
Podgląd informacji o pojazdach	Wykonalne	Pojazdy
Dodawanie/Modyfikowanie/Usuwanie informacji o pojazdach	Wykonalne	Pojazdy
Podgląd informacji o pojazdach jako ciężarówkach	Wykonalne	Pojazdy, Ciezarówki
Dodawanie/Modyfikowanie/Usuwanie informacji o pojazdach jako ciężarówkach	Wykonalne	Pojazdy, Ciezarówki
Podgląd informacji o pojazdach jako wózkach widłowych	Wykonalne	Pojazdy, Wozki_widlowe
Dodawanie/Modyfikowanie/Usuwanie informacji o pojazdach jako wózkach widłowych	Wykonalne	Pojazdy, Wozki_widlowe
Podgląd informacji o magazynach	Wykonalne	Magazyny, Adresy
Dodawanie/Modyfikowanie/Usuwanie informacji o magazynach	Wykonalne	Magazyny, Adresy
Podgląd informacji o klientach	Wykonalne	Hurtownie, Klienci
Dodawanie/Modyfikowanie/Usuwanie informacji o klientach	Wykonalne	Hurtownie, Klienci
Podgląd informacji o pracownikach	Wykonalne	Hurtownie, Adresy, Pracownicy
Dodawanie/Modyfikowanie/Usuwanie informacji o pracownikach	Wykonalne	Hurtownie, Adresy, Pracownicy
Podgląd informacji o pracownikach jako magazynierach	Wykonalne	Pracownicy, Magazynierze
Dodawanie/Modyfikowanie/Usuwanie informacji o pracownikach jako magazynierach	Wykonalne	Pracownicy, Magazynierze
Podgląd informacji o pracownikach jako kierowcach	Wykonalne	Pracownicy, Kierowcy
Dodawanie/Modyfikowanie/Usuwanie informacji o pracownikach jako kierowcach	Wykonalne	Pracownicy, Kierowcy
Podgląd informacji o produktach	Wykonalne	Produkty
Dodawanie/Modyfikowanie/Usuwanie informacji o produktach	Wykonalne	Produkty
Podgląd informacji o pojazdach obsługiwanych przez kierowców	Wykonalne	Pojazd, Kierowca, Przejazdy
Dodawanie/Modyfikowanie/Usuwanie informacji o pojazdach obsługiwanych przez kierowców	Wykonalne	Pojazd, Kierowca, Przejazdy

Tabela 13: **Klucze kandydujące i główne**

Transakcja	Weryfikacja wykonalności	Potrzebne dane
Podgląd informacji o magazynach w których pracują magazynierzy	Wykonalne	Magazyn, Magazynierzy, Zmiany
Dodawanie/Modyfikowanie/Usuwanie informacji o pojazdach obsługiwanych przez kierowców	Wykonalne	Magazyn, Magazynierzy, Zmiany
Podgląd informacji o magazynach przechowujących produkty	Wykonalne	Magazyn, Produkt, Palety
Dodawanie/Modyfikowanie/Usuwanie informacji o pojazdach obsługiwanych przez kierowców	Wykonalne	Magazyn, Produkt, Palety
Podgląd informacji o produktach kupionych przez klientów	Wykonalne	Klient, Produkt, Zamowienia
Dodawanie/Modyfikowanie/Usuwanie informacji o produktach kupionych przez klientów	Wykonalne	Klient, Produkt, Zamowienia

Tabela 14: **Klucze kandydujące i główne**

5.2 Strojenie bazy danych – dobór indeksów

Do strojenia bazy danych zostały wykorzystane następujące indeksy. Indeksy zostały wygenerowane automatycznie wraz z generacją kodu SQL:

- IX_adres_hurtowni - wyszukiwanie adresu hurtowni,
- IX_hurtownia_zatrudnia_pracownika - wyszukiwanie hurtowni, która zatrudnia pracownika,
- IX_adres_pracownika - wyszukiwanie adresu pracownika,
- IX_stanowisko_pracownika - wyszukiwanie stanowiska pracownika,
- IX_pojazd_hurtownia - wyszukiwanie hurtowni, do której należy pojazd,
- IX_model_pojazd - wyszukiwanie modelu pojazdu,
- IX_magazyn_hurtowni - wyszukiwanie hurtowni, do której należy magazyn,
- IX_adres_magazynu - wyszukiwanie adresu magazynu,
- IX_adres_klienta - wyszukiwanie adresu klienta,
- IX_poczta_adresu - wyszukiwanie poczty adresu,
- IX_wlasciciel_hurtowni - wyszukiwanie hurtowni, która należy do właściciela
- IX_adres_wlasciciela - wyszukiwanie adresu właściciela,
- IX_model_marka - wyszukiwanie marki modelu
- IX_hurtownia_klienta - wyszukiwanie hurtowni, do której jest przypisany klient
- IX_hurtownia_produktu - wyszukiwanie hurtowni, do której jest przypisany produkt

5.3 Skrypt SQL zakładający bazę danych

```
/*
Created: 22.11.2023
Modified: 10.12.2023
Project: Hurtownia czesci elektronicznych
Model: Hurtownia Czesci Elektronicznych
Company: Politechnika Warszawska - WEITI
Author: Anh Quan Do, Agata Zatorska
Version: 1.0
Database: Oracle 18c
*/


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

-- Table Hurtownie

CREATE TABLE Hurtownie(
  Nr_hurtowni Integer NOT NULL,
  Nazwa Varchar2(50 ) NOT NULL,
  Data_zalozenia Date NOT NULL,
  Nr_adresu Integer NOT NULL
)
/

-- Create indexes for table Hurtownie

CREATE INDEX IX_adres_hurtowni ON Hurtownie (Nr_adresu)
/

-- Add keys for table Hurtownie

ALTER TABLE Hurtownie ADD CONSTRAINT Unique_Identifier1 PRIMARY KEY (Nr_hurtowni)
/

-- Table Pracownicy

CREATE TABLE Pracownicy(
  Nr_pracownika Integer NOT NULL,
  Imie Varchar2(30 ) NOT NULL,
  Nazwisko Varchar2(50 ) NOT NULL,
  Data_urodzenia Date NOT NULL,
  PESEL Char(11 ),
  Plec Char(1 ) NOT NULL
       CHECK (Plec IN('K','M')),
  Data_zatrudnienia Date NOT NULL,
  Nr_konta Char(34 ),
  Email Varchar2(30 ),
  Nr_telefonu Varchar2(14 ),
  Wynagrodzenie Integer NOT NULL,
  Nr_hurtowni Integer NOT NULL,
  Nr_adresu Integer NOT NULL,
  Nr_stanowiska Integer NOT NULL
)
/

-- Create indexes for table Pracownicy

CREATE INDEX IX_hurtownia_zatrudnia_pracownika ON Pracownicy (Nr_hurtowni)
/

CREATE INDEX IX_adres_pracownika ON Pracownicy (Nr_adresu)
/

CREATE INDEX IX_stanowisko_pracownika ON Pracownicy (Nr_stanowiska)
/

-- Add keys for table Pracownicy

ALTER TABLE Pracownicy ADD CONSTRAINT Unique_Identifier2 PRIMARY KEY (Nr_pracownika)
/

-- Table Kierowcy
```

```
CREATE TABLE Kierowcy(
  Nr_pracownika Integer NOT NULL,
  Nr_prawa_jazdy Char(13 ) NOT NULL,
  Data_waznosci_prawa_jazdy Date NOT NULL,
  Kod_kategorii Char(1 ) NOT NULL
        CHECK (Kod_kategorii IN('A','B','C','D','T'))
)
/

-- Add keys for table Kierowcy

ALTER TABLE Kierowcy ADD CONSTRAINT Unique_Identifier3 PRIMARY KEY (Nr_pracownika)
/

-- Table Magazynierzy

CREATE TABLE Magazynierzy(
  Nr_pracownika Integer NOT NULL,
  Szkolenie_BHP Char(1 ) NOT NULL
        CHECK (Szkolenie_BHP IN('T','N')),
  Uprawnienia_wozek Char(1 ) NOT NULL
        CHECK (Uprawnienia_wozek IN ('T','N'))
)
/

-- Add keys for table Magazynierzy

ALTER TABLE Magazynierzy ADD CONSTRAINT Unique_Identifier4 PRIMARY KEY (Nr_pracownika)
/

-- Table Pojazdy

CREATE TABLE Pojazdy(
  Nr_pojazdu Integer NOT NULL,
  Nr_rejestracyjny Varchar2(10 ),
  VIN Char(17 ) NOT NULL,
  Pojemnosc_silnika Integer NOT NULL,
  Rodzaj_paliwa Char(15 ) NOT NULL
        CHECK (Rodzaj_paliwa IN('LPG','b95','b98','diesel','elektryczny')),
  Nr_hurtowni Integer NOT NULL,
  Nr_modelu Integer NOT NULL
)
/

-- Create indexes for table Pojazdy

CREATE INDEX IX_pojazd_hurtowni ON Pojazdy (Nr_hurtowni)
/

CREATE INDEX IX_marka_pojazd ON Pojazdy (Nr_modelu)
/

-- Add keys for table Pojazdy

ALTER TABLE Pojazdy ADD CONSTRAINT Unique_Identifier5 PRIMARY KEY (Nr_pojazdu)
/

-- Table Wozki_widlowe

CREATE TABLE Wozki_widlowe(
  Nr_pojazdu Integer NOT NULL,
  Pojemnosc_udzwigu Integer NOT NULL,
  Wysokosc_unoszenia Integer NOT NULL
)
/

-- Add keys for table Wozki_widlowe

ALTER TABLE Wozki_widlowe ADD CONSTRAINT Unique_Identifier6 PRIMARY KEY (Nr_pojazdu)
/

-- Table Ciezarowki

CREATE TABLE Ciezarowki(
  Nr_pojazdu Integer NOT NULL,
```

```
  Pojemnosc Integer NOT NULL,
  Typ_ciezarowki Char(15 ) NOT NULL
        CHECK (Typ_ciezarowki IN('furgon','chlodnia','kontener','plandeka','siodĆowy'))
)
/

-- Add keys for table Ciezarowki

ALTER TABLE Ciezarowki ADD CONSTRAINT Unique_Identifier7 PRIMARY KEY (Nr_pojazdu)
/

-- Table Magazyny

CREATE TABLE Magazyny(
  Nr_magazynu Integer NOT NULL,
  Powierzchnia Integer NOT NULL,
  Typ_magazynu Char(15 ) NOT NULL
        CHECK (Typ_magazynu IN('Chlodnia','Sezonowy','Cross-Docking','Centralny','Dystrybucyjny')),
  Nazwa Varchar2(30 ) NOT NULL,
  Pojemnosc Integer NOT NULL,
  Nr_hurtowni Integer NOT NULL,
  Nr_adresu Integer NOT NULL
)
/

-- Create indexes for table Magazyny

CREATE INDEX IX_magazyn_hurtowni ON Magazyny (Nr_hurtowni)
/

CREATE INDEX IX_adres_magazynu ON Magazyny (Nr_adresu)
/

-- Add keys for table Magazyny

ALTER TABLE Magazyny ADD CONSTRAINT Unique_Identifier8 PRIMARY KEY (Nr_magazynu)
/

-- Table Produkty

CREATE TABLE Produkty(
  Nr_produktu Integer NOT NULL,
  Nazwa Varchar2(50 ) NOT NULL,
  Opis Varchar2(300 ),
  Ilosc_produktow Integer NOT NULL,
  Producent Varchar2(30 ) NOT NULL,
  Cena Integer NOT NULL,
  Nr_hurtowni Integer NOT NULL
)
/

-- Create indexes for table Produkty

CREATE INDEX IX_hurtownia_produktu ON Produkty (Nr_hurtowni)
/

-- Add keys for table Produkty

ALTER TABLE Produkty ADD CONSTRAINT Unique_Identifier9 PRIMARY KEY (Nr_produktu)
/

-- Table Klienci

CREATE TABLE Klienci(
  Nr_klienta Integer NOT NULL,
  Imie Varchar2(30 ) NOT NULL,
  Nazwisko Varchar2(50 ) NOT NULL,
  PESEL Char(11 ),
  Data_urodzenia Date,
  Plec Char(1 )
        CHECK (Plec IN('K','M')),
  Email Varchar2(50 ),
  Nr_telefonu Char(14 ),
  Nr_adresu Integer,
  Nr_hurtowni Integer NOT NULL
```

```
)
/

-- Create indexes for table Klienci

CREATE INDEX IX_adres_klienta ON Klienci (Nr_adresu)
/

CREATE INDEX IX_hurtownia_klienta ON Klienci (Nr_hurtowni)
/

-- Add keys for table Klienci

ALTER TABLE Klienci ADD CONSTRAINT Unique_Identifier10 PRIMARY KEY (Nr_klienta)
/

-- Table Zmiany

CREATE TABLE Zmiany(
  Nr_pracownika Integer NOT NULL,
  Nr_magazynu Integer NOT NULL
)
/

-- Table Palety

CREATE TABLE Palety(
  Nr_magazynu Integer NOT NULL,
  Nr_produktu Integer NOT NULL
)
/

-- Table Zamowienia

CREATE TABLE Zamowienia(
  Nr_klienta Integer NOT NULL,
  Nr_produktu Integer NOT NULL
)
/

-- Table Przejazdy

CREATE TABLE Przejazdy(
  Nr_pojazdu Integer NOT NULL,
  Nr_pracownika Integer NOT NULL
)
/

-- Table Adresy

CREATE TABLE Adresy(
  Nr_adresu Integer NOT NULL,
  Miasto Varchar2(100 ) NOT NULL,
  Ulica Varchar2(150 ) NOT NULL,
  Nr_budynku Integer NOT NULL,
  Nr_lokalu Integer ,
  Nr_poczty Integer NOT NULL
)
/

-- Create indexes for table Adresy

CREATE INDEX IX_poczta_adresu ON Adresy (Nr_poczty)
/

-- Add keys for table Adresy

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

-- Table Wlasciciele

CREATE TABLE Wlasciciele(
  Nr_wlasciciela Integer NOT NULL,
  Imie Varchar2(30 ) NOT NULL,
```

```
  Nazwisko Varchar2(50 ) NOT NULL,
  Data_urodzenia Date NOT NULL,
  PESEL Char(11 ),
  Plec Char(1 ) NOT NULL
       CHECK (Plec IN('K','M')),
  Nr_konta Char(34 ),
  Email Varchar2(30 ),
  Wynagrodzenie Integer NOT NULL,
  Nr_hurtowni Integer NOT NULL,
  Nr_adresu Integer NOT NULL
)
/

-- Create indexes for table Wlasciciele

CREATE INDEX IX_wlasciciel_hurtowni ON Wlasciciele (Nr_hurtowni)
/

CREATE INDEX IX_adres_wlasciciela ON Wlasciciele (Nr_adresu)
/

-- Add keys for table Wlasciciele

ALTER TABLE Wlasciciele ADD CONSTRAINT PK_Wlasciciele PRIMARY KEY (Nr_wlasciciela)
/

-- Table Marki

CREATE TABLE Marki(
  Nr_marki Integer NOT NULL,
  Nazwa Varchar2(100 ) NOT NULL,
  Data_zalozenia Date NOT NULL
)
/

-- Add keys for table Marki

ALTER TABLE Marki ADD CONSTRAINT PK_Marki PRIMARY KEY (Nr_marki)
/

-- Table Modele

CREATE TABLE Modele(
  Nr_modelu Integer NOT NULL,
  Nazwa Varchar2(100 ) NOT NULL,
  Rok_produkcji Char(4 ) NOT NULL,
  Nr_marki Integer NOT NULL
)
/

-- Create indexes for table Modele

CREATE INDEX IX_model_marka ON Modele (Nr_marki)
/

-- Add keys for table Modele

ALTER TABLE Modele ADD CONSTRAINT PK_Modele PRIMARY KEY (Nr_modelu)
/

-- Table Stanowiska

CREATE TABLE Stanowiska(
  Nr_stanowiska Integer NOT NULL,
  Nazwa Varchar2(100 ) NOT NULL,
  Opis Varchar2(500 ) NOT NULL
)
/

-- Add keys for table Stanowiska

ALTER TABLE Stanowiska ADD CONSTRAINT PK_Stanowiska PRIMARY KEY (Nr_stanowiska)
/

-- Table Poczty
```

```
CREATE TABLE Poczty(
   Nr_poczty Integer NOT NULL,
   Kod_pocztowy Char(6 ) NOT NULL,
   Poczta Varchar2(30 ) NOT NULL
)
/

-- Add keys for table Poczty

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

ALTER TABLE Poczty ADD CONSTRAINT Kod_pocztowy UNIQUE (Kod_pocztowy)
/

-- Table and Columns comments section

COMMENT ON COLUMN Poczty.Kod_pocztowy IS 'format␣XX-XXX'
/

-- Trigger for sequence HurtowniaSeq1 for column Nr_hurtowni in table Hurtownie ---------
CREATE OR REPLACE TRIGGER ts_Hurtownie_HurtowniaSeq1 BEFORE INSERT
ON Hurtownie FOR EACH ROW
BEGIN
   :new.Nr_hurtowni := HurtowniaSeq1.nextval;
END;
/
CREATE OR REPLACE TRIGGER tsu_Hurtownie_HurtowniaSeq1 AFTER UPDATE OF Nr_hurtowni
ON Hurtownie FOR EACH ROW
BEGIN
   RAISE_APPLICATION_ERROR(-20010,'Cannot␣update␣column␣Nr_hurtowni␣in␣table␣Hurtownie␣as␣it␣uses␣sequence.');
END;
/

-- Trigger for sequence HurtowniaSeq1 for column Nr_pracownika in table Pracownicy ---------
CREATE OR REPLACE TRIGGER ts_Pracownicy_HurtowniaSeq1 BEFORE INSERT
ON Pracownicy FOR EACH ROW
BEGIN
   :new.Nr_pracownika := HurtowniaSeq1.nextval;
END;
/
CREATE OR REPLACE TRIGGER tsu_Pracownicy_HurtowniaSeq1 AFTER UPDATE OF Nr_pracownika
ON Pracownicy FOR EACH ROW
BEGIN
   RAISE_APPLICATION_ERROR(-20010,'Cannot␣update␣column␣Nr_pracownika␣in␣table␣Pracownicy␣as␣it␣uses␣sequence.');
END;
/

-- Trigger for sequence HurtowniaSeq1 for column Nr_pojazdu in table Pojazdy ---------
CREATE OR REPLACE TRIGGER ts_Pojazdy_HurtowniaSeq1 BEFORE INSERT
ON Pojazdy FOR EACH ROW
BEGIN
   :new.Nr_pojazdu := HurtowniaSeq1.nextval;
END;
/
CREATE OR REPLACE TRIGGER tsu_Pojazdy_HurtowniaSeq1 AFTER UPDATE OF Nr_pojazdu
ON Pojazdy FOR EACH ROW
BEGIN
   RAISE_APPLICATION_ERROR(-20010,'Cannot␣update␣column␣Nr_pojazdu␣in␣table␣Pojazdy␣as␣it␣uses␣sequence.');
END;
/

-- Trigger for sequence HurtowniaSeq1 for column Nr_magazynu in table Magazyny ---------
CREATE OR REPLACE TRIGGER ts_Magazyny_HurtowniaSeq1 BEFORE INSERT
ON Magazyny FOR EACH ROW
BEGIN
   :new.Nr_magazynu := HurtowniaSeq1.nextval;
END;
/
CREATE OR REPLACE TRIGGER tsu_Magazyny_HurtowniaSeq1 AFTER UPDATE OF Nr_magazynu
ON Magazyny FOR EACH ROW
BEGIN
   RAISE_APPLICATION_ERROR(-20010,'Cannot␣update␣column␣Nr_magazynu␣in␣table␣Magazyny␣as␣it␣uses␣sequence.');
END;
```

```
/

-- Trigger for sequence HurtowniaSeq1 for column Nr_produktu in table Produkty ---------
CREATE OR REPLACE TRIGGER ts_Produkty_HurtowniaSeq1 BEFORE INSERT
ON Produkty FOR EACH ROW
BEGIN
  :new.Nr_produktu := HurtowniaSeq1.nextval;
END;
/
CREATE OR REPLACE TRIGGER tsu_Produkty_HurtowniaSeq1 AFTER UPDATE OF Nr_produktu
ON Produkty FOR EACH ROW
BEGIN
  RAISE_APPLICATION_ERROR(-20010,'Cannot update column Nr_produktu in table Produkty as it uses sequence.');
END;
/

-- Trigger for sequence HurtowniaSeq1 for column Nr_klienta in table Klienci ---------
CREATE OR REPLACE TRIGGER ts_Klienci_HurtowniaSeq1 BEFORE INSERT
ON Klienci FOR EACH ROW
BEGIN
  :new.Nr_klienta := HurtowniaSeq1.nextval;
END;
/
CREATE OR REPLACE TRIGGER tsu_Klienci_HurtowniaSeq1 AFTER UPDATE OF Nr_klienta
ON Klienci FOR EACH ROW
BEGIN
  RAISE_APPLICATION_ERROR(-20010,'Cannot update column Nr_klienta in table Klienci as it uses sequence.');
END;
/

-- Trigger for sequence HurtowniaSeq1 for column Nr_adresu in table Adresy ---------
CREATE OR REPLACE TRIGGER ts_Adresy_HurtowniaSeq1 BEFORE INSERT
ON Adresy FOR EACH ROW
BEGIN
  :new.Nr_adresu := HurtowniaSeq1.nextval;
END;
/
CREATE OR REPLACE TRIGGER tsu_Adresy_HurtowniaSeq1 AFTER UPDATE OF Nr_adresu
ON Adresy FOR EACH ROW
BEGIN
  RAISE_APPLICATION_ERROR(-20010,'Cannot update column Nr_adresu in table Adresy as it uses sequence.');
END;
/

-- Trigger for sequence HurtowniaSeq1 for column Nr_wlasciciela in table Wlasciciele ---------
CREATE OR REPLACE TRIGGER ts_Wlasciciele_HurtowniaSeq1 BEFORE INSERT
ON Wlasciciele FOR EACH ROW
BEGIN
  :new.Nr_wlasciciela := HurtowniaSeq1.nextval;
END;
/
CREATE OR REPLACE TRIGGER tsu_Wlasciciele_HurtowniaSeq1 AFTER UPDATE OF Nr_wlasciciela
ON Wlasciciele FOR EACH ROW
BEGIN
  RAISE_APPLICATION_ERROR(-20010,'Cannot update column Nr_wlasciciela in table Wlasciciele as it uses sequence.');
END;
/

-- Trigger for sequence HurtowniaSeq1 for column Nr_marki in table Marki ---------
CREATE OR REPLACE TRIGGER ts_Marki_HurtowniaSeq1 BEFORE INSERT
ON Marki FOR EACH ROW
BEGIN
  :new.Nr_marki := HurtowniaSeq1.nextval;
END;
/
CREATE OR REPLACE TRIGGER tsu_Marki_HurtowniaSeq1 AFTER UPDATE OF Nr_marki
ON Marki FOR EACH ROW
BEGIN
  RAISE_APPLICATION_ERROR(-20010,'Cannot update column Nr_marki in table Marki as it uses sequence.');
END;
/

-- Trigger for sequence HurtowniaSeq1 for column Nr_modelu in table Modele ---------
CREATE OR REPLACE TRIGGER ts_Modele_HurtowniaSeq1 BEFORE INSERT
ON Modele FOR EACH ROW
```

```
BEGIN
   :new.Nr_modelu := HurtowniaSeq1.nextval;
END;
/
CREATE OR REPLACE TRIGGER tsu_Modele_HurtowniaSeq1 AFTER UPDATE OF Nr_modelu
ON Modele FOR EACH ROW
BEGIN
   RAISE_APPLICATION_ERROR(-20010,'Cannot update column Nr_modelu in table Modele as it uses sequence.');
END;
/

-- Trigger for sequence HurtowniaSeq1 for column Nr_stanowiska in table Stanowiska ---------
CREATE OR REPLACE TRIGGER ts_Stanowiska_HurtowniaSeq1 BEFORE INSERT
ON Stanowiska FOR EACH ROW
BEGIN
   :new.Nr_stanowiska := HurtowniaSeq1.nextval;
END;
/
CREATE OR REPLACE TRIGGER tsu_Stanowiska_HurtowniaSeq1 AFTER UPDATE OF Nr_stanowiska
ON Stanowiska FOR EACH ROW
BEGIN
   RAISE_APPLICATION_ERROR(-20010,'Cannot update column Nr_stanowiska in table Stanowiska as it uses sequence.');
END;
/

-- Trigger for sequence HurtowniaSeq1 for column Nr_poczty in table Poczty ---------
CREATE OR REPLACE TRIGGER ts_Poczty_HurtowniaSeq1 BEFORE INSERT
ON Poczty FOR EACH ROW
BEGIN
   :new.Nr_poczty := HurtowniaSeq1.nextval;
END;
/
CREATE OR REPLACE TRIGGER tsu_Poczty_HurtowniaSeq1 AFTER UPDATE OF Nr_poczty
ON Poczty FOR EACH ROW
BEGIN
   RAISE_APPLICATION_ERROR(-20010,'Cannot update column Nr_poczty in table Poczty as it uses sequence.');
END;
/


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

ALTER TABLE Pracownicy ADD CONSTRAINT Zatrudnia FOREIGN KEY (Nr_hurtowni) REFERENCES Hurtownie (Nr_hurtowni)
/



ALTER TABLE Pojazdy ADD CONSTRAINT Posiada_pojazd FOREIGN KEY (Nr_hurtowni) REFERENCES Hurtownie (Nr_hurtowni)
/



ALTER TABLE Magazyny ADD CONSTRAINT Posiada_magazyn FOREIGN KEY (Nr_hurtowni) REFERENCES Hurtownie (Nr_hurtowni)
/



ALTER TABLE Wlasciciele ADD CONSTRAINT Wlasciciel_posiada_hurtownie FOREIGN KEY (Nr_hurtowni) REFERENCES Hurtownie
/



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



ALTER TABLE Pracownicy ADD CONSTRAINT Adres_jest_przypisany_do_pracownika FOREIGN KEY (Nr_adresu) REFERENCES Adresy
/



ALTER TABLE Magazyny ADD CONSTRAINT Magazyn_jest_zlokalizowany FOREIGN KEY (Nr_adresu) REFERENCES Adresy (Nr_adresu
/
```

```
ALTER TABLE Hurtownie ADD CONSTRAINT Adres_jest_przypisany_do_hurtowni FOREIGN KEY (Nr_adresu) REFERENCES Adresy (N
/

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

ALTER TABLE Pracownicy ADD CONSTRAINT Pracownik_obsadza_stanowisko FOREIGN KEY (Nr_stanowiska) REFERENCES Stanowisk
/

ALTER TABLE Wlasciciele ADD CONSTRAINT Adres_jest_przypisany_do_wlasciciela FOREIGN KEY (Nr_adresu) REFERENCES Adre
/

ALTER TABLE Pojazdy ADD CONSTRAINT Pojazd_posiada_model FOREIGN KEY (Nr_modelu) REFERENCES Modele (Nr_modelu)
/

ALTER TABLE Modele ADD CONSTRAINT Model_nalezy_do_marki FOREIGN KEY (Nr_marki) REFERENCES Marki (Nr_marki)
/

ALTER TABLE Produkty ADD CONSTRAINT Sprzedaje_produkty FOREIGN KEY (Nr_hurtowni) REFERENCES Hurtownie (Nr_hurtowni)
/

ALTER TABLE Klienci ADD CONSTRAINT Posiada_klientow FOREIGN KEY (Nr_hurtowni) REFERENCES Hurtownie (Nr_hurtowni)
/
```

5.4 Przykłady zapytań i poleceń SQL odnoszących się do bazy danych

- Uporządkowanie pracowników według daty zatrudnienia (od najdłużej do najkrócej pracującego),

```
SELECT NR_PRACOWNIKA, IMIE, NAZWISKO, DATA_ZATRUDNIENIA
FROM PRACOWNICY
ORDER BY DATA_ZATRUDNIENIA ASC;
```

	NR_PRACOWNIKA	IMIE	NAZWISKO	DATA_ZATRUDNIENIA
1	142	Joanna	Kocioł	10/05/27
2	143	Stefan	Komoda	14/06/05
3	144	Danuta	Bolko	17/01/06
4	141	Janusz	Brych	20/05/06

- Informacja o ilości mężczyzn pośród klientów,

```
SELECT COUNT(*) AS LICZBA_MEZCZYZN_WSROD_KLIENTOW
FROM KLIENCI
WHERE PLEC='M';
```

	LICZBA_MEZCZYZN_WSROD_KLIENTOW
1	2

- Informacja o tym, które z adresów posiadają pocztę w Puławach,

```
SELECT MIASTO, ULICA, NR_BUDYNKU, NR_LOKALU
FROM ADRESY
WHERE NR_POCZTY = 102;
```

	MIASTO	ULICA	NR_BUDYNKU	NR_LOKALU
1	Puławy	Żeromskiego	11	30
2	Puławy	Baca	5	(null)

- Uporządkowanie pracowników, którzy zarabiają ponad 4500zł, według wynagrodzenia (od najniższego do najwyższego),

```
SELECT IMIE, NAZWISKO, WYNAGRODZENIE FROM PRACOWNICY
WHERE WYNAGRODZENIE > 4500
ORDER BY WYNAGRODZENIE ASC;
```

	IMIE	NAZWISKO	WYNAGRODZENIE
1	Janusz	Brych	5000
2	Danuta	Bolko	7000
3	Joanna	Kocioł	9000

- Informacja o sumie ilości produktów marek Ladex i Kondex na stanie hurtowni.

```
SELECT SUM(ILOSC_PRODUKTOW) AS Suma_Produktow_Firm_Ledox_Kondex
FROM PRODUKTY
WHERE PRODUCENT IN ('Ledox', 'Kondex');
```

	SUMA_PRODUKTOW_FIRM_LEDOX_KONDEX
1	65000

6 Bibliografia

Podczas realizacji projektu korzystaliśmy z materiałów stworzonych i dostarczonych nam przez prowadzącego - dr hab. inż Marcina Kowalczyka.