Kacper Bloch
Kamil Dzierżanowski

Projekt WBD

Salon samochodowy - “Sprzedam Opla” Dealership

Spis treści

1. Zakres i cel projektu (opis założeń funkcjonalnych bazy)

Celem projektu jest zaprojektowanie, na poziomie konceptualnym oraz logicznym, a także fizyczna implementacja relacyjnej bazy danych.

Baza danych umożliwia obsługę i zarządzanie salonem samochodowym. Utworzona baza danych będzie oparta o rozwiązanie firmy Oracle.

Oprogramowanie użyte podczas realizacji projektu:
Toad Data Modeler
Sql Developer-4.1.5.21.78-x64
Oracle Database 12c

1.1 Założenia funkcjonalne

(z przyczyn technicznych, w języku angielskim)

A **car dealership** is a single business venue that offers **cars** for sale to the **customers**. It is a partnership with a certain carmaker, therefore the cars offered are of that brand. Cars can be **new and used**. New cars are **ordered** from a **factory** either by a request from a client or by a decision of the management. A factory provides **technical details** and a **suggested retail price** about each **model** of a car. Each car for sale is supervised by a **salesman**. Salesmen are further supervised by **managers**. Every employee receives **salaries** for their work, usually once per month. Customers can apply for a **leasing plan** offered by one of the **lessors**. Customers can also take out an **insurance**.

2. Definicja systemu

Wyróżnione funkcjonalności systemu

1. Podgląd danych personalnych pracowników.
2. Podgląd danych personalnych właściciela konta.
3. Modyfikacja/dodawanie/usuwanie danych personalnych pracowników.
4. Podgląd danych personalnych klientów.
5. Modyfikacja/wprowadzanie/usuwanie danych personalnych klientów.
6. Podgląd historii leasingów, ubezpieczeń, samochodów zakupionych przez użytkownika.
7. Modyfikacja/wprowadzanie/usuwanie danych leasingów, ubezpieczeń, samochodów.
8. Podgląd danych samochodów, leasingów, ubezpieczeń.
9. Podgląd listy fabryk produkujących samochody.
10. Modyfikacja/usuwanie/dodawanie fabryk.
11. Podgląd/modyfikacja/dodawanie/usuwanie dodatkowego wyposażenia samochodów.
12. Podgląd listy wynagrodzeń.
13. Podgląd listy leasingodawców.
14. Modyfikacja/usuwanie/dodawanie leasingodawców do bazy.

2.1 Perspektywy użytkowników

2.1.1 Administrator

Administrator ma dostęp do wszystkich funkcjonalności systemu oraz modyfikacji struktury bazy danych. Ma on uprawnienia administratora bazy danych Oracle.

2.2.2 Właściciel salonu

Właściciel może zobaczyć wszystkie dane przechowywane w bazie danych.

2.2.3 Księgowa

Odpowiada za finanse - ma dostęp do danych personalnych pracowników, informacji o ich wynagrodzeniu, sprzedanych i kupionych samochodach.

2.2.4 Klient

Klient ma dostęp do swoich danych, może je modyfikować. Ma podgląd do historii swoich transakcji - kupionych i sprzedanych samochodów, swoich ubezpieczeń i leasingów.

2.2.5 Manager

Manager zarządza pracownikami. Ma dostęp do ich danych, wynagrodzeń, może je modyfikować. Ma dostęp do danych klientów, leasingów, ubezpieczeń i samochodów.

2.2.6 Sprzedawca

Ma dostęp do swoich danych, może je modyfikować. Ma dostęp do danych samochodów, może je modyfikować.

2.2.7 Inni pracownicy

Mają dostęp do swoich danych i mogą je modyfikować.

3. Model konceptualny

3.1 Definicja zbiorów encji określonych w projekcie oraz określenie atrybutów i ich dziedzin (decyzje projektowe)

Przy przekonwertowaniu modelu na model bazy Oracle typ danych może ulec zmianie. Typ Money zostanie zamieniony na Number(10,2), SmallInt na Integer, VarChar na VarChar2.

Encja Dealership - określa strukturę i definiuje salon samochodowy

Caption	**Name**	**Data Type**	**Data Type Param 1**	**Primary Identifier**	**Mandatory**	**Description**
dealership_id	dealership_id	SmallInt		true	true	Id of dealership
name	name	VarChar(%p1)	30	false	true	Name of dealership
owner	owner	VarChar(%p1)	60	false	true	First and last name of owner
founding_date	founding_date	Date		false	false	Date when dealership was founded

square_meters_[m2]	square_meters_[m2]	Number(8,2)		false	false	Usable floor area in square meters [m^2]
phone_number	phone_number	VarChar(%p1)	15	false	false	Dealership's phone number
email	email	VarChar(%p1)	30	false	false	Dealership's email adress

Encja Adress - określa strukturę i definiuje adres

Caption	**Name**	**Data Type**	**Data Type Param 1**	**Primary Identifier**	**Mandatory**	**Description**
adress_id	adress_id	SmallInt		true	true	ID of adress in database
street	street	VarChar(%p1)	30	false	false	Name of street
number	number	VarChar(%p1)	10	false	true	Number of building
apartment	apartment	VarChar(%p1)	10	false	false	Number of apartment
city	city	VarChar(%p1)	30	false	true	City
postal	postal	VarChar(%p1)	6	false	true	Postal adress

Encja Factory - określa strukturę i definiuje fabrykę samochodów, do której można kierować zamówienia

Caption	Name	Data Type	Data Type Param 1	Primary Identifier	Mandatory	Description
factory_id	factory_id	SmallInt		true	true	ID of factory in database
name	name	VarChar(%p1)	30	false	true	Name of factory
phone_number	phone_number	VarChar(%p1)	15	false	false	Factory's phone number
email	email	VarChar(%p1)	30	false	false	Factory's email adress

Encja Lessor - określa strukturę i definiuje instytucję finansową, która jest partnerem salonu samochodowego i oferuje leasing klientom

Caption	Name	Data Type	Data Type Param 1	Primary Identifier	Mandatory	Description
lessor_id	lessor_id	SmallInt		true	true	ID of lessor in database
name	name	VarChar(%p1)	30	false	true	Name of lessor
phone_number	phone_number	VarChar(%p1)	15	false	false	Lessor's phone number
email	email	VarChar(%p1)	30	false	false	Lessor's email adress

Encja Car - określa strukturę i definiuje samochód

Caption	Name	Data Type	Data Type Param 1	Primary Identifier	Mandatory	Description
car_id	car_id	SmallInt		true	true	ID of car in database
is_new	is_new	Boolean		false	true	Determine if car is new
is_delivered	is_delivered	Boolean		false	true	Determine if car is delivered to delearship
VIN	VIN	VarChar(%p1)	16	false	true	VIN number of car
production_year	production_year	Number(4,0)		false	true	Production year of car
color	color	VarChar(%p1)	30	false	true	Color of car
price	price	Money		false	false	Car's price
is_4x4	is_4x4	Boolean		false	true	Determine if car has 4x4 drive
transmission	transmission	VarChar(%p1)	30	false	true	Determine car's transmission
engine	engine	VarChar(%p1)	30	false	true	Type of car's engine
is_reserved	is_reserved	Boolean		false	true	Determine if car is reserved

Encja Model - określa strukturę i definiuje model samochodu

Caption	Name	Data Type	Data Type Param 1	Primary Identifier	Mandatory	Description
model_id	model_id	SmallInt		true	true	ID of model in database
name	name	VarChar(%p1)	30	false	true	Name
generation	generation	VarChar(%p1)	10	false	false	Generation of model

Encja Brand - określa strukturę i definiuje markę samochodu

Caption	Name	Data Type	Data Type Param 1	Primary Identifier	Mandatory	Description
brand_id	brand_id	SmallInt		true	true	ID of brand in database
name	name	VarChar(%p1)	30	false	true	Name of brand

Encja employee - określa strukturę i definiuje pracownika. Ze względu na ograniczoną liczbę płci dopuszczalne wartości atrybutu gender zostały ograniczone do “F” i “M”

Caption	Name	Data Type	Data Type Param 1	Primary Identifier	Mandatory	Description
employee_id	employee_id	SmallInt		true	true	ID of employee in database
first_name	first_name	VarChar(%p1)	30	false	true	First name of employee
last_name	last_name	VarChar(%p1)	30	false	true	Last name of employee

basic_salary	basic_salary	Money		false	true	Employee's basic salary
date_employed	date_employed	Date		false	true	Date when person was hired
contract_expiration	contract_expiration	Date		false	false	Date till employee is hired
gender	gender	Character(%p1)	1	false	true	Gender - male or female. Male - M, female - F
phone_number	phone_number	VarChar(%p1)	15	false	false	Personal phone number of employee
email	email	VarChar(%p1)	30	false	false	Personal email adress of employee

Encja insurance - określa strukturę i definiuje ubezpieczenie

Caption	**Name**	**Data Type**	**Data Type Param 1**	**Primary Identifier**	**Mandatory**	**Description**
insurance_id	insurance_id	SmallInt		true	true	ID of insurance in database
description	description	VarChar(%p1)	200	false	true	Short description
date_from	date_from	Date		false	false	Date when insurance was taken out

date_to	date_to	Date		false	false	Date when insurance expires
type	type	VarChar(%p1)	20	false	true	Type

Encja Leasing - określa strukturę i definiuje leasing

Caption	Name	Data Type	Data Type Param 1	Primary Identifier	Mandatory	Description
leasing_id	leasing_id	SmallInt		true	true	ID of leasing in database
desctription	desctription	VarChar(%p1)	200	false	true	Short desc.
date_from	date_from	Date		false	true	Date when leasing was taken out
date_to	date_to	Date		false	true	Date when leasing expires
amount	amount	Money		false	true	Value of leasing
instalment	instalment	Money		false	true	Monthly instalment

Encja Salary - określa strukturę i definiuje wynagrodzenie

Caption	Name	Data Type	Data Type Param 1	Primary Identifier	Mandatory	Description
salary_id	salary_id	SmallInt		true	true	ID of salary in database
date	date	Date		false	true	Date when salary was paid
amount	amount	Money		false	true	Amount of money

description	description	VarChar(%p1)	200	false	false	Short description and additional informations
additional_amount	additional_amount	Money		false	false	Amount of additional money eg. bonuses

Encja Client - określa strukturę i definiuje klienta salonu. Ze względu na ograniczoną liczbę płci dopuszczalne wartości atrybutu gender został ograniczone do “F” i “M”

Caption	**Name**	**Data Type**	**Data Type Param 1**	**Primary Identifier**	**Mandatory**	**Description**
id_client	id_client	SmallInt		true	true	ID of client in database
last_name	last_name	VarChar(%p1)	30	false	true	Last name of client
first_name	first_name	VarChar(%p1)	30	false	true	First name of client
gender	gender	Character(%p1)	1	false	true	Gender of client - male - M
phone_number	phone_number	VarChar*%p1)	15	false	false	Personal phone number
email	email	VarChar(%p1)	30	false	false	Personal email
type_of_personal_document	type_of_personal_document	VarChar(%p1)	30	false	false	Type of client's personal document
number_of_personal_document	number_of_personal_document	VarChar(%p1)	30	false	false	Number of client's personal

						document

Encja Additional_Equipment - określa strukturę i definiuje dodatkowe wyposażenie samochodu

Caption	Name	Data Type	Data Type Param 1	Primary Identifier	Mandatory	Description
additional_equipment_id	additional_equipment_id	SmallInt		true	true	ID of additional equipment in database
name	name	VarChar(%p1)	30	false	true	Name of equipment
price	price	Money		false	false	Equipment's price
description	description	VarChar(%p1)	200	false	false	Short description

Encja Profession - określa strukturę i definiuje stanowisko pracownika

Caption	Name	Data Type	Data Type Param 1	Primary Identifier	Mandatory	Description
profession_id	profession_id	SmallInt		true	true	ID of profession in database
name	name	VarChar(%p1)	30	false	true	Name of profession
description	description	VarChar(%p1)	200	false	false	Short description of responsibilities.

Encja Deal - określa strukturę i definiuje transakcje zakupu samochodu

Caption	Name	Data type	Data type param 1	Primary Identifier	Mandatory	Description
Deal_id	Deal_id	Integer		true	true	Deal's ID in database
Price	Price	Money		false	true	Value of deal
Date	Date	Date		false	true	Date of signing
Description	Description	VarChar(%p1)	200	false	false	Short description, extra information

3.2 Ustalenie związków i ich typów między encjami (decyzje projektowe)

Większość związków jest typu 1:n, jednak istnieją również inne m.in związki encji Adress z innymi encjami są typu 1:1, gdyż każdy obiekt może mieć tylko 1 adres, a dany adres może być przypisany tylko do 1 obiektu. Zaprojektowane zostały również związki n:m np. związek między encjami Car oraz Additonal_equipment, gdyż samochód może mieć wiele różnych typów dodatków, a dodatki mogą znajdować się w różnych samochodach.

Caption	Name	Parent Entity	Child Entity	Cardinality	Degree
has_car_produced	has_car_produced	Dealership	Factory	1..n - 0..m	binary
are_partners	are_partners	Dealership	Lessor	1..n - 0..m	binary
sells_car	sells_car	Dealership	Car	1..1 - 0..m	binary
factory_has_adress	factory_has_adress	Adress	Factory	1..1 - 0..1	binary
is_a_specific_model	is_a_specific_model	Car	Model	0..n - 1..1	binary
is_a_specific_brand	is_a_specific_brand	Model	Brand	0..n - 1..1	binary

has_employee	has_employee	Dealership	Employee	1..1 - 0..m	binary
offers_insurance	offers_insurance	Dealership	Insurance	1..1 - 0..m	binary
lessor_has_adress	lessor_has_adress	Lessor	Adress	0..1 - 1..1	binary
lessor_offers_leasing	lessor_offers_leasing	Lessor	Leasing	1..1 - 0..m	binary
offers_leasing	offers_leasing	Leasing	Dealership	0..n - 1..1	binary
earns	earns	Employee	Salary	1..1 - 0..m	binary
employee_has_adress	employee_has_adress	Employee	Adress	0..1 - 1..1	binary
dealership_has_adress	dealership_has_adress	Dealership	Adress	0..1 - 1..1	binary
has_client	has_client	Dealership	Client	1..1 - 0..m	binary
pays_for	pays_for	Client	Insurance	1..1 - 0..m	binary
is_a_lessee	is_a_lessee	Client	Leasing	1..1 - 0..m	binary
supervises	supervises	Employee	Employee	1..1 - 0..m	unary
car_has_additional_equipment	car_has_additional_equipment	Car	Additional_equipment	1..n - 0..m	binary
is_being_sold_by	is_being_sold_by	Employee	Car	1..1 - 0..m	binary
takes_care_of	takes_care_of	Employee	Insurance	1..1 - 0..m	binary
employee's profession	employee's profession	Employee	Profession	0..n - 1..1	binary
bought_car	bought_car	Client	Deal	1..1 - 0..m	binary
sold_car	sold_car	Dealership	Deal	1..1 - 0..m	binary
was_sold	was_sold	Car	Deal	1..1 - 0..1	binary

3.3 Klucze kandydujące i główne

Zdecydowaliśmy się na użycie jako kluczy tylko sztucznie wygenerowanych numerów ID, takie podejście zwiększa czytelność bazy, wprowadza jednolite standardy oraz przyspiesza działanie bazy np. wyszukiwanie rekordów, gdyż wartości id są z mniejszego zakresu wartości niż inne klucze kandydujące np.numer VIN.

Klucze główne

Encja	**Klucz główny**	**Inne klucze kandydujące**
Dealership	dealership_id	name
Adress	adress_id	-
Factory	factory_id	name
Lessor	lessor_id	name
Car	car_id	VIN
Model	model_id	-
Brand	brand_id	name
Employee	employee_id	PESEL
Insurance	insurance_id	-
Leasing	leasing_id	-
Salary	salary_id	-
Client	id_client	PESEL
Additional_equipment	additional_equipment_id	name
Profession	profession_id	name
Deal	deal_id	-

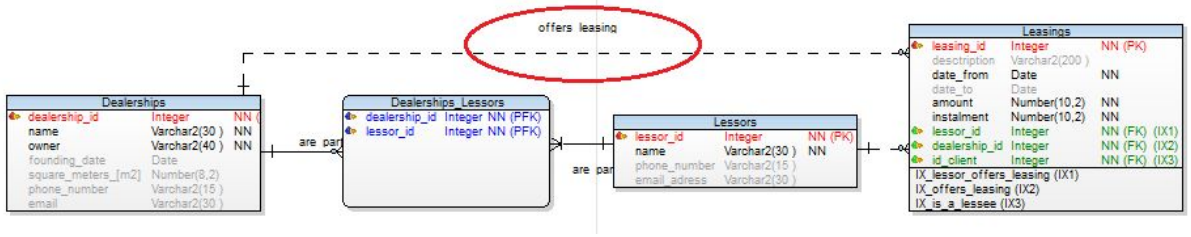
3.4 Schemat ER na poziomie konceptualnym

↳ Załącznik nr 1

3.5 Problem pułapek szczelinowych i wachlarzowych – analiza i przykłady

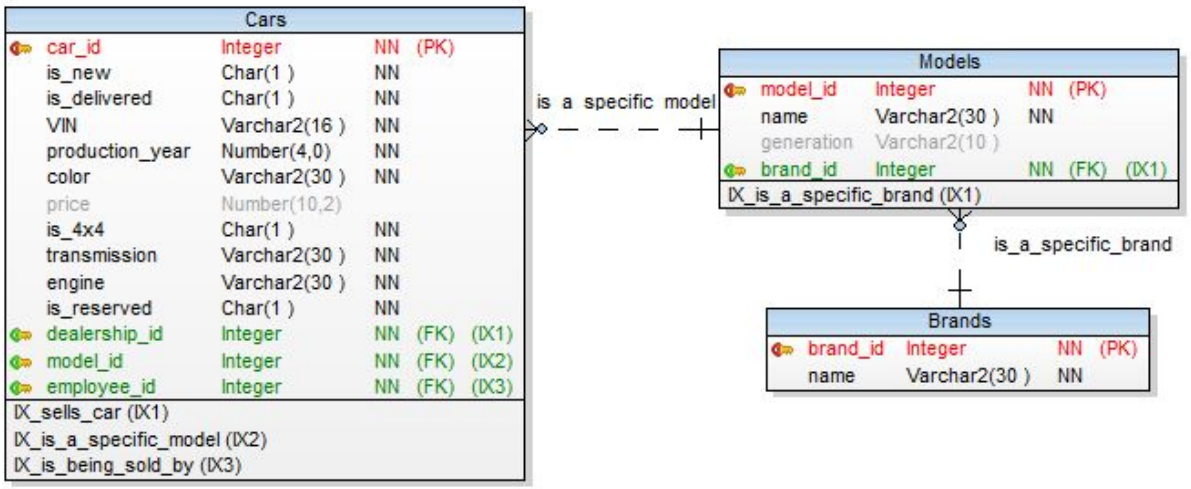
Szczelinowa

Powstało ryzyko, że jeżeli każdy leasing będzie uzależniony od leasingodawcy, to po ich usunięciu znikną te leasingi. Rozwiązaniem tego problemu stało się połączenie leasingów z salonem:



Wachlarzowa

Pułapka ta ujawniła się w relacjach `Car >- Brands -< Models` - każda marka może być przypisana do wielu samochodów, ale też oferować wiele modeli, przez co nie można byłoby ustalić, jaki jest model samochodu. Rozwiązaniem jest przypisanie marki do modelu, i modelu do samochodu:



4. Model logiczny

4.1 Usunięcie właściwości niekompatybilnych z modelem relacyjnym - przykłady

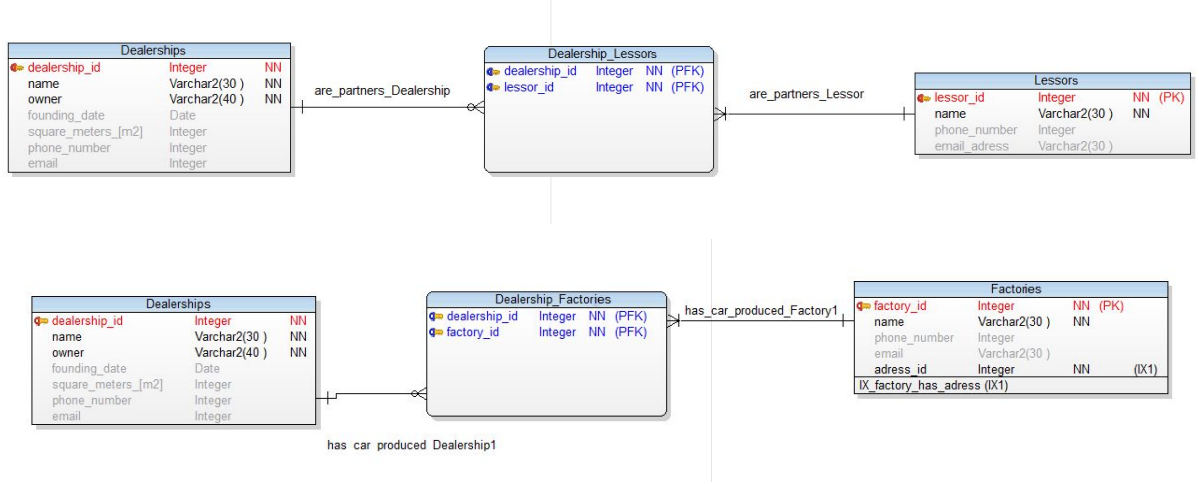
W celu przekształcenia modelu konceptualnego w logiczny model relacyjny, usunęliśmy niekompatybilności modelu konceptualnego z modelem relacyjnym - zastąpiliśmy związki wiele do wielu tablicami bridge'ującymi. Dla każdej encji w diagramie stworzyliśmy tabelę. Nazwę każdej encji zastąpiliśmy liczbą mnogą w celu odróżnienia relacji od encji. Identyfikujący atrybut encji stał się kluczem głównym tabeli. Wszystkie inne atrybuty encji stały się niegłównymi atrybutami tabeli.Dla każdego związku „jeden do wiele" wstawiliśmy klucz główny tabeli ze strony jednej linii związku. Do tabeli reprezentującej stronę wiele linii związku wstawiliśmy klucz obcy. Opcjonalność na stronie wiele linii związku określiłą czy klucz obcy może być null, czy nie. Występujący związek rekurencyjny jeden do wielu przekształcony został w jedną tabelę z kluczem obcym, którego wartościami są wartości klucza głównego.

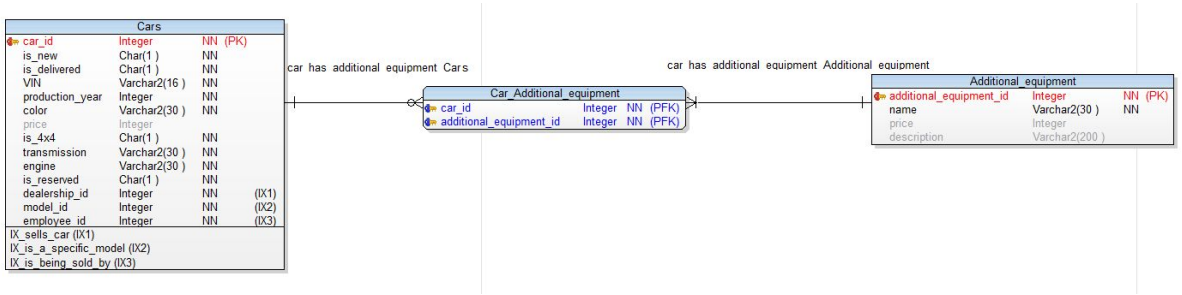
W modelu konceptualnym znajdują się 3 związki n:m. W modelu relacyjnym zostały one zastąpione tablicami bridge'ującymi.

Poniższe związki są związkami n:m:

- has_car_produced
- are_partners
- car_has_additional_equipment

Struktura po usunięciu tych związków:





Na powyższych przykładach widać również wymienione wcześniej zmiany.

4.2 Proces normalizacji - analiza i przykłady

1PN

- ➔ Relacja jest w pierwszej postaci normalnej, jeśli każda wartość atrybutu w każdej krotce tej relacji jest **wartością elementarną**, czyli nierozkładalną
- ➔ Relacja jest w pierwszej postaci normalnej, jeśli nie ma **powtarzających się grup**

Przykład:
Cars

car_id	is_new	is_delivered	is_reserved	VIN	production year	price	Other information	Model
230	true	true	false	ABC123456789DEF	2017	100000	2.0 Turbo Diesel, Red, 4x4, automatic,	Opel Astra

Takie podejście nie pozwala nam na wyszukanie samochodów ze względu na konkretną jego cechę np. silnika. Również nie możemy wylistować wszystkich samochodów marki Opel.

Cars po modyfikacji

car_id	is_new	is_delivered	is_reserved	VIN	production year	price	Engine	Color	Transmission	is 4x4	Brand	Model
230	true	true	false	ABC123456789DEF	2017	100000	2.0 Turbo Diesel	Red	Automatic	true,	Opel	Astra

Inny przykład (w stosunku do projektu konceptualnego część atrybutów w tej sekcji została pominięta, aby zwiększyć przejrzystość):

Employees

employee_id	first_name	last_name	profession	dealership_name	adress_id	owner	phone_number
1	Patryk	Gozdera	Salesman	Forgut	34	Kacper Bloch	555666777
2	Tomek	Kurowski	Manager	Forgut	34	Kacper Bloch	555666777
3	Michał	Białocerkowiec	Manager	Rotyl	12	Kamil Dzierżanowski	111222333
4	Paweł	Kowalik	Salesman	Rotyl	12	Kamil Dzierżanowski	111222333

W powyższym przykładzie atrybut owner składa się z imienia i nazwiska, więc teoretycznie nie jest wartością atomową, ale projekt nie przewiduje sytuacji, gdzie konieczne jest pobranie imienia lub nazwiska osobno. Z tego powodu imię i nazwisko właściciela traktujemy jako jedną wartość.

Jak widać występują powtarzające się grupy

Po normalizacji:

Employees

employee_id	first_name	last_name	profession_id	dealership_id
1	Patryk	Gozdera	1	1
2	Tomek	Kurowski	2	1
3	Michał	Białocerkowiec	2	2
4	Paweł	Kowalik	1	2

Dealerships i Professions

dealership_id	dealership_name	adress_id	owner	phone_number
1	Forgut	34	Kacper Bloch	555666777
2	Rotyl	12	Kamil Dzierżanowski	111222333

profession_id	name
1	Salesman
2	Manager

2PN

- ➔ Relacja jest w drugiej postaci normalnej, jeśli jest w 1PN
- ➔ Każdy atrybut tej relacji **nie wchodzący** w skład żadnego klucza potencjalnego jest w **pełni funkcyjnie zależny** od wszystkich kluczy potencjalnych tej relacji
- ➔ Relacja jest w 2PN jeżeli każdy atrybut nie wchodzący w skład klucza **zależy od klucza a nie od jego części**

W zaprojektowanej bazie wszystkie klucze potencjalne są kluczami prostymi, a więc 2PN została osiągnięta.

3PN

- ➔ Relacja jest w trzeciej postaci normalnej, jeśli jest w 2PN
- ➔ **Wszystkie niekluczowe kolumny są określone kluczem, całym kluczem i tylko kluczem**

Przykład:

Cars

car_id	is_new	is_delivered	is_reserved	VIN	production_year	price	engine	color	transmission	is_4x4	brand_id	model_id
230	true	true	false	ABC123456789DEF	2017	100000	2.0 Turbo Diesel	Red	Automatic	true,	2	1

Marka samochodu wynika bezpośrednio z modelu.

Po zmianie:

Cars

car_id	is_new	is_delivered	is_reserved	VIN	production_year	price	engine	color	transmission	is_4x4	model_id
230	true	true	false	ABC123456789DEF	2017	100000	2.0 Turbo Diesel	Red	Automatic	true,	1

brand_id	name
1	Ford
2	Opel
3	Fiat

model_id	name	generation	brand_id
1	Vectra	IV	2
2	Astra	II	2
3	Focus	III	1

Przy projektowaniu bazy staraliśmy się aby baza była w 3PN. Między innymi dla wszelkich powtarzających się grup powstały dodatkowe encje, które w modelu relacyjnym stały się tablicami np. zamiast atrybutu Brand w encji Car powstała dodatkowa encja Brand. Pola wielowartościowe i segmentowe nie zostały zaprojektowane. Przy projektowaniu sporo problemów przysparza adres, dlatego postanowiliśmy, że będzie on osobną encją. Wszelkie klucze potencjalne są kluczami prostymi a niegłówne atrybuty nie są przechodnio zależne od kluczy potencjalnych.

4.3 Schemat ER na poziomie modelu logicznego

↬ Załącznik nr 2

4.4 Więzy integralności

Zadbaliśmy żeby wszystkie pola były polami atomowymi. Wszystkie klucze oznaczone są jako UNIQUE. Wszystkie atrybuty, których brak miałby wpływ na działanie bazy danych mają nałożone ograniczenie NOT NULL.

4.5 Proces denormalizacji - analiza i przykłady

→ Cars

car_id	is_new	is_delivered	is_reserved	VIN	production year	price	Engine_id	Color_id	Transmission_id	is 4x4	Model_id
230	true	true	false	ABC123456789DEF	2017	100000	2	3	1	true	2
231	true	true	false	AAA123456789DEF	2016	90000	2	3	1	false	2

Engines

engine_id	name
1	1.6 Diesel
2	2.0 Turbo Diesel

Colors

color_id	name
1	Blue
2	Black
3	Red
4	Green

Transmissions

transmission_id	name
1	Automatic
2	6-Manual

Atrybuty samochodu takie jak silnik, kolor, skrzynia biegów dla wielu samochodów często się powtarzają, więc dla tych atrybutów powinny istnieć stworzyć osobne relacje.

Stworzenie osobnych relacji znacząco spowolniłoby działanie bazy, gdyż najczęstszym zapytaniem do bazy jest wylistowanie samochodów razem z jego parametrami, a takie zapytanie wymagałoby łączenia kilku tabel naraz.

Zwykle klienci szukają samochodu konkretnego modelu, a nie koloru, więc uznaliśmy, że relacja dla modelu pozostanie, aby uniknąć problemów przy wyszukiwaniu np. literówka.

Podsumowując zrezygnowaliśmy z tabel Engines, Colors, Transmissions

→ Zastanawialiśmy się nad zastąpieniem tabeli Adresses poprzez wielosegmentowy atrybut, ale taka zmiana spowodowałaby zmniejszenie elastyczności bazy np. nie moglibyśmy zmienić nazwy ulicy bez ingerencji w cały adres.

→ Atrybut owner w tabeli Dealerships jest polem segmentowym (zawiera imię i nazwisko), gdyż projekt bazy nie przewiduje konieczności wyciągnięcia pojedynczego elementu (imienia bądź nazwiska) z bazy.

5. Faza fizyczna

5.1 Projekt transakcji i weryfikacja ich wykonalności

Transakcja	Potrzebne zasoby	Czy wykonalne?	Uwagi
Podgląd danych personalnych pracowników	Employees, Addresses	Tak	-
Podgląd danych personalnych właściciela konta	Clients	Tak	-
Modyfikacja/dodawanie/usuwanie danych personalnych pracowników	Employees, Addresses	Tak	-
Podgląd danych personalnych klientów	Clients	Tak	-
Modyfikacja/wprowadzanie/usuwanie danych personalnych klientów	Clients	Tak	-
Podgląd historii leasingów, ubezpieczeń, samochodów zakupionych przez użytkownika	Clients, Leasings, Insurances, Deals	Tak	-
Modyfikacja/wprowadzanie/usuwanie danych leasingów, ubezpieczeń, samochodów	Clients, Leasings, Insurances, Deals	Tak	-
Podgląd danych samochodów, leasingów, ubezpieczeń	Cars, Leasings, Insurances	Tak	-
Podgląd listy fabryk produkujących samochody	Factories	Tak	-
Modyfikacja/usuwanie/dodawanie fabryk	Factories	Tak	-
Podgląd/modyfikacja/dodawanie/usuwanie dodatkowego wyposażenia samochodów	Additional_equipment, Car_Additional_equipment	Tak	-
Podgląd listy wynagrodzeń	Salaries	Tak	-

Podgląd listy leasingodawców	Lessors	Tak	-
Modyfikacja/usuwanie/dodawanie leasingodawców do bazy	Lessors	Tak	-

5.2 Strojenie bazy danych - dobór indeksów

Samochody - salon macierzysty, model i nadzorca

```
CREATE INDEX IX_sells_car ON Cars (dealership_id)
CREATE INDEX IX_is_a_specific_model ON Cars (model_id)
CREATE INDEX IX_is_being_sold_by ON Cars (employee_id)
```

Modele - marka

```
CREATE INDEX IX_is_a_specific_brand ON Models (brand_id)
```

Pracownicy - miejsce pracy, przełożony, specjalność

```
CREATE INDEX IX_has_employee ON Employees (dealership_id)
CREATE INDEX IX_supervises ON Employees (FK_employee_id)
CREATE INDEX IX_employees profession ON Employees (profession_id)
```

Ubezpieczenia - salon oferujący, ubezpieczany, nadzorca

```
CREATE INDEX IX_offers_insurance ON Insurances (dealership_id)
CREATE INDEX IX_pays_for ON Insurances (id_client)
CREATE INDEX IX_takes_care_of ON Insurances (employee_id)
```

Leasingi - leasingodawca, salon oferujący, leasingbiorca

```
CREATE INDEX IX_lessor_offers_leasing ON Leasings (lessor_id)
CREATE INDEX IX_offers_leasing ON Leasings (dealership_id)
CREATE INDEX IX_is_a_lessee ON Leasings (id_client)
```

Wypłaty - pracownik

```
CREATE INDEX IX_earns ON Salaries (employee_id)
```

Klienci - salon

```
CREATE INDEX IX_has_client ON Clients (dealership_id)
```

5.3 Skrypt SQL zakładający bazę danych

```
CREATE TABLE Dealerships(
  dealership_id Integer NOT NULL,
  name Varchar2(30 ) NOT NULL,
  owner Varchar2(60 ) NOT NULL,
  founding_date Date,
  square_meters_[m2] Number(8,2),
  phone_number Varchar2(15 ),
  email Varchar2(30 )
)
/

CREATE TABLE Addresses(
  adress_id Integer NOT NULL,
  street Varchar2(30 ),
  number Varchar2(10 ) NOT NULL,
  apartment Varchar2(10 ),
  city Varchar2(30 ) NOT NULL,
  postal Varchar2(6 ) NOT NULL,
  lessor_id Integer,
  employee_id Integer,
  dealership_id Integer
)
/

CREATE INDEX IX_lessor_has_adress ON Addresses (lessor_id)
/

CREATE INDEX IX_employee_has_adress ON Addresses (employee_id)
/

CREATE INDEX IX_dealership_has_adress ON Addresses (dealership_id)
/

ALTER TABLE Addresses ADD CONSTRAINT Unique_Identifier2 PRIMARY KEY (adress_id)
/

CREATE TABLE Factories(
  factory_id Integer NOT NULL,
  name Varchar2(30 ) NOT NULL,
  phone_number Varchar2(15 ),
  email Varchar2(30 ),
  adress_id Integer NOT NULL
)
/

CREATE INDEX IX_factory_has_adress ON Factories (adress_id)
/

ALTER TABLE Factories ADD CONSTRAINT Unique_Identifier3 PRIMARY KEY (factory_id)
/

CREATE TABLE Lessors(
  lessor_id Integer NOT NULL,
  name Varchar2(30 ) NOT NULL,
  phone_number Varchar2(15 ),
```

```
  email_adress Varchar2(30 )
)
/

ALTER TABLE Lessors ADD CONSTRAINT Unique_Identifier4 PRIMARY KEY (lessor_id)
/

CREATE TABLE Cars(
  car_id Integer NOT NULL,
  is_new Char(1 ) NOT NULL,
  is_delivered Char(1 ) NOT NULL,
  VIN Varchar2(16 ) NOT NULL,
  production_year Number(4,0) NOT NULL,
  color Varchar2(30 ) NOT NULL,
  price Number(10,2),
  is_4x4 Char(1 ) NOT NULL,
  transmission Varchar2(30 ) NOT NULL
       CONSTRAINT ValidValuestransmission CHECK ((transmission IN
('Automatic','Manual'))),
  engine Varchar2(30 ) NOT NULL,
  is_reserved Char(1 ) NOT NULL,
  is_sold Char(1 ) NOT NULL,
  dealership_id Integer NOT NULL,
  model_id Integer NOT NULL,
  employee_id Integer NOT NULL
)
/

CREATE INDEX IX_sells_car ON Cars (dealership_id)
/

CREATE INDEX IX_is_a_specific_model ON Cars (model_id)
/

CREATE INDEX IX_is_being_sold_by ON Cars (employee_id)
/

ALTER TABLE Cars ADD CONSTRAINT Unique_Identifier5 PRIMARY KEY (car_id)
/

CREATE TABLE Models(
  model_id Integer NOT NULL,
  name Varchar2(30 ) NOT NULL,
  generation Varchar2(10 ),
  brand_id Integer NOT NULL
)
/

CREATE INDEX IX_is_a_specific_brand ON Models (brand_id)
/

ALTER TABLE Models ADD CONSTRAINT Unique_Identifier6 PRIMARY KEY (model_id)
/

CREATE TABLE Brands(
  brand_id Integer NOT NULL,
  name Varchar2(30 ) NOT NULL
)
/
```

```
ALTER TABLE Brands ADD CONSTRAINT Unique_Identifier7 PRIMARY KEY (brand_id)
/

CREATE TABLE Employees(
  employee_id Integer NOT NULL,
  first_name Varchar2(30 ) NOT NULL,
  last_name Varchar2(30 ) NOT NULL,
  basic_salary Number(10,2) NOT NULL,
  date_employed Date NOT NULL,
  contract_expiration Date,
  gender Char(1 ) NOT NULL
       CONSTRAINT CheckConstraintA1 CHECK (--IN ('M', 'F')),
  phone_number Varchar2(15 ),
  email Varchar2(30 ),
  dealership_id Integer NOT NULL,
  FK_employee_id Integer NOT NULL,
  profession_id Integer NOT NULL
)
/

CREATE INDEX IX_has_employee ON Employees (dealership_id)
/

CREATE INDEX IX_supervises ON Employees (FK_employee_id)
/

CREATE INDEX IX_employee's profession ON Employees (profession_id)
/

ALTER TABLE Employees ADD CONSTRAINT Unique_Identifier8 PRIMARY KEY (employee_id)
/

CREATE TABLE Insurances(
  insurance_id Integer NOT NULL,
  description Varchar2(200 ) NOT NULL,
  date_from Date,
  date_to Date,
  type Varchar2(20 ) NOT NULL,
  dealership_id Integer NOT NULL,
  id_client Integer NOT NULL,
  employee_id Integer NOT NULL
)
/

CREATE INDEX IX_offers_insurance ON Insurances (dealership_id)
/

CREATE INDEX IX_pays_for ON Insurances (id_client)
/

CREATE INDEX IX_takes_care_of ON Insurances (employee_id)
/

ALTER TABLE Insurances ADD CONSTRAINT Unique_Identifier10 PRIMARY KEY (insurance_id)
/

CREATE TABLE Leasings(
  leasing_id Integer NOT NULL,
```

```
  desctription Varchar2(200 ),
  date_from Date NOT NULL,
  date_to Date,
  amount Number(10,2) NOT NULL,
  instalment Number(10,2) NOT NULL,
  lessor_id Integer NOT NULL,
  dealership_id Integer NOT NULL,
  id_client Integer NOT NULL
)
/

CREATE INDEX IX_lessor_offers_leasing ON Leasings (lessor_id)
/

CREATE INDEX IX_offers_leasing ON Leasings (dealership_id)
/

CREATE INDEX IX_is_a_lessee ON Leasings (id_client)
/

ALTER TABLE Leasings ADD CONSTRAINT Unique_Identifier11 PRIMARY KEY (leasing_id)
/

CREATE TABLE Salaries(
  salary_id Integer NOT NULL,
  date Date NOT NULL,
  amount Number(10,2) NOT NULL,
  description Varchar2(200 ),
  additional_amount Number(10,2),
  employee_id Integer NOT NULL
)
/

CREATE INDEX IX_earns ON Salaries (employee_id)
/

ALTER TABLE Salaries ADD CONSTRAINT Unique_Identifier13 PRIMARY KEY (salary_id)
/

CREATE TABLE Clients(
  id_client Integer NOT NULL,
  last_name Varchar2(30 ) NOT NULL,
  first_name Varchar2(30 ) NOT NULL,
  gender Char(1 ) NOT NULL
       CONSTRAINT CheckConstraintA1 CHECK (--IN ('M', 'F')),
  phone_number Varchar2(15 ),
  email Varchar2(30 ),
  type_of_personal_document Varchar2(30 ),
  number_of_personal_document Varchar2(30 ),
  dealership_id Integer NOT NULL
)
/

CREATE INDEX IX_has_client ON Clients (dealership_id)
/

ALTER TABLE Clients ADD CONSTRAINT Unique_Identifier14 PRIMARY KEY (id_client)
/
```

```
CREATE TABLE Additional_equipment(
  additional_equipment_id Integer NOT NULL,
  name Varchar2(30 ) NOT NULL,
  price Number(10,2),
  description Varchar2(200 )
)
/

ALTER TABLE Additional_equipment ADD CONSTRAINT Unique_Identifier15 PRIMARY KEY
(additional_equipment_id)
/

CREATE TABLE Professions(
  profession_id Integer NOT NULL,
  name Varchar2(30 ) NOT NULL,
  description Varchar2(200 )
)
/

ALTER TABLE Professions ADD CONSTRAINT Unique_Identifier16 PRIMARY KEY (profession_id)
/

CREATE TABLE Dealerships_Factories(
  dealership_id Integer NOT NULL,
  factory_id Integer NOT NULL
)
/

CREATE TABLE Dealerships_Lessors(
  dealership_id Integer NOT NULL,
  lessor_id Integer NOT NULL
)
/

CREATE TABLE Car_Additional_equipment(
  car_id Integer NOT NULL,
  additional_equipment_id Integer NOT NULL
)
/

CREATE TABLE Deals(
  Deal_id Integer NOT NULL,
  Price Number(10,2) NOT NULL,
  Date Date NOT NULL,
  Description Varchar2(200 ),
  id_client Integer NOT NULL,
  car_id Integer NOT NULL,
  dealership_id Integer NOT NULL
)
/

CREATE INDEX IX_Relationship1 ON Deals (id_client)
/

CREATE INDEX IX_Relationship2 ON Deals (car_id)
/

CREATE INDEX IX_Relationship3 ON Deals (dealership_id)
/
```

```
ALTER TABLE Deals ADD CONSTRAINT Key7 PRIMARY KEY (Deal_id)
/

ALTER TABLE Cars ADD CONSTRAINT sells_car FOREIGN KEY (dealership_id) REFERENCES
Dealerships (dealership_id)
/

ALTER TABLE Factories ADD CONSTRAINT factory_has_adress FOREIGN KEY (adress_id) REFERENCES
Addresses (adress_id)
/

ALTER TABLE Cars ADD CONSTRAINT is_a_specific_model FOREIGN KEY (model_id) REFERENCES
Models (model_id)
/

ALTER TABLE Models ADD CONSTRAINT is_a_specific_brand FOREIGN KEY (brand_id) REFERENCES
Brands (brand_id)
/

ALTER TABLE Employees ADD CONSTRAINT has_employee FOREIGN KEY (dealership_id) REFERENCES
Dealerships (dealership_id)
/

ALTER TABLE Insurances ADD CONSTRAINT offers_insurance FOREIGN KEY (dealership_id)
REFERENCES Dealerships (dealership_id)
/

ALTER TABLE Addresses ADD CONSTRAINT lessor_has_adress FOREIGN KEY (lessor_id) REFERENCES
Lessors (lessor_id)
/

ALTER TABLE Leasings ADD CONSTRAINT lessor_offers_leasing FOREIGN KEY (lessor_id)
REFERENCES Lessors (lessor_id)
/

ALTER TABLE Leasings ADD CONSTRAINT offers_leasing FOREIGN KEY (dealership_id) REFERENCES
Dealerships (dealership_id)
/

ALTER TABLE Salaries ADD CONSTRAINT earns FOREIGN KEY (employee_id) REFERENCES Employees
(employee_id)
/

ALTER TABLE Addresses ADD CONSTRAINT employee_has_adress FOREIGN KEY (employee_id)
REFERENCES Employees (employee_id)
/

ALTER TABLE Addresses ADD CONSTRAINT dealership_has_adress FOREIGN KEY (dealership_id)
REFERENCES Dealerships (dealership_id)
/

ALTER TABLE Clients ADD CONSTRAINT has_client FOREIGN KEY (dealership_id) REFERENCES
Dealerships (dealership_id)
/

ALTER TABLE Insurances ADD CONSTRAINT pays_for FOREIGN KEY (id_client) REFERENCES Clients
(id_client)
/
```

```
ALTER TABLE Leasings ADD CONSTRAINT is_a_lessee FOREIGN KEY (id_client) REFERENCES Clients
(id_client)
/

ALTER TABLE Employees ADD CONSTRAINT supervises FOREIGN KEY (FK_employee_id) REFERENCES
Employees (employee_id)
/

ALTER TABLE Cars ADD CONSTRAINT is_being_sold_by FOREIGN KEY (employee_id) REFERENCES
Employees (employee_id)
/

ALTER TABLE Insurances ADD CONSTRAINT takes_care_of FOREIGN KEY (employee_id) REFERENCES
Employees (employee_id)
/

ALTER TABLE Employees ADD CONSTRAINT employee's profession FOREIGN KEY (profession_id)
REFERENCES Professions (profession_id)
/

ALTER TABLE Deals ADD CONSTRAINT bought_car FOREIGN KEY (id_client) REFERENCES Clients
(id_client)
/

ALTER TABLE Deals ADD CONSTRAINT was_sold FOREIGN KEY (car_id) REFERENCES Cars (car_id)
/

ALTER TABLE Deals ADD CONSTRAINT sold_car FOREIGN KEY (dealership_id) REFERENCES
Dealerships (dealership_id)
/
```

5.4 Przykłady zapytań i poleceń SQL odnoszących się do bazy danych

Znajdź wszystkie samochody oferowane przez salon o podanej nazwie

```
SELECT * FROM Cars
JOIN Dealerships ON Cars.dealership_id = Dealerships.dealership_id
WHERE Dealerships.name = 'Autosalon Ursynów sp. z o. o.';
```

Uzyskaj kontakt do człowieka odpowiedzialnego za samochód o podanym ID

```
SELECT first_name, last_name, phone_number, email FROM Employees
JOIN Cars ON Employees.employee_id = Cars.employee_id
WHERE Cars.car_id IS 1234;
```

Znajdź salony w Warszawie

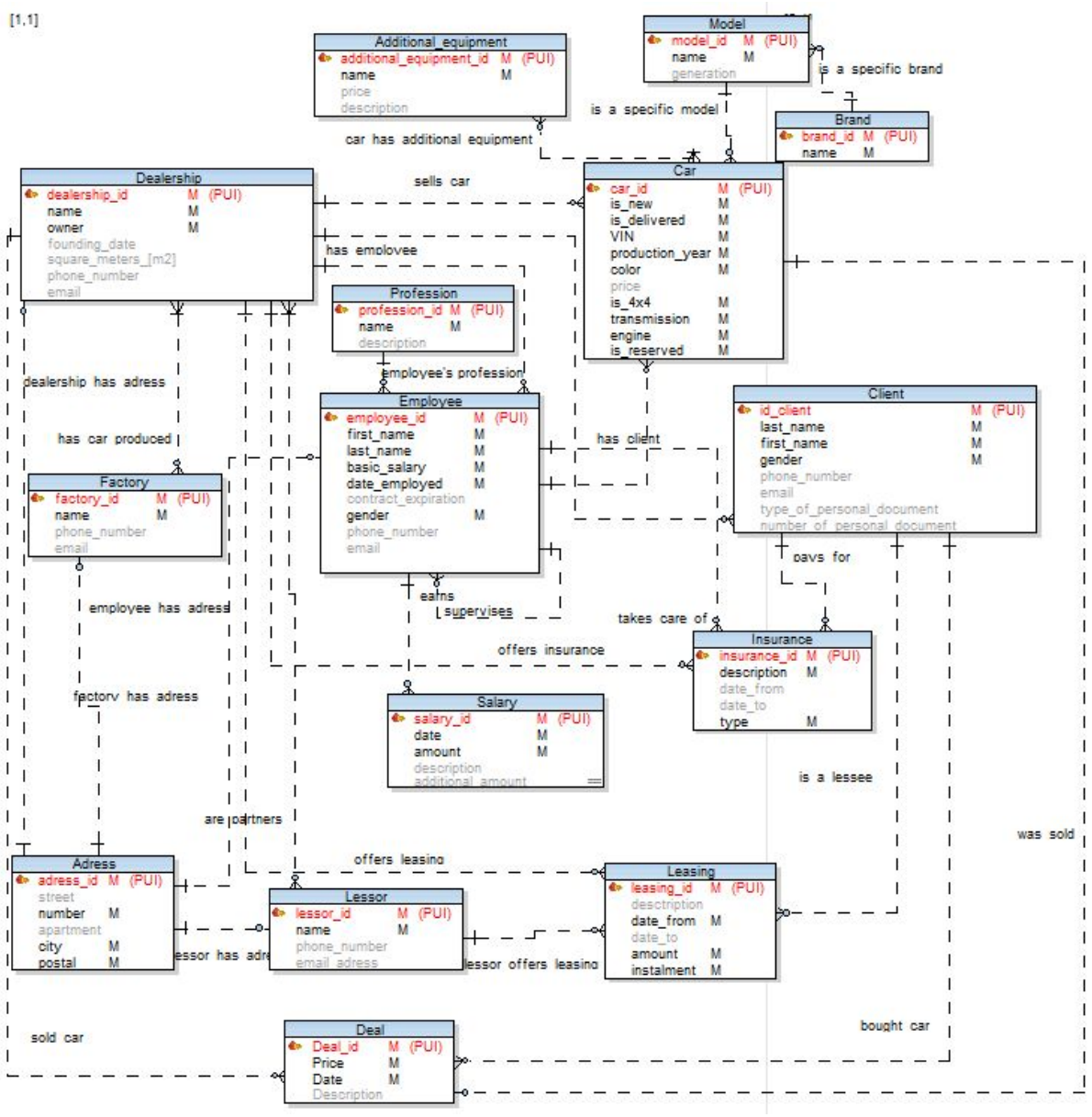
```
SELECT name, phone_number, email FROM Dealerships
JOIN Addresses ON Dealerships.dealership_id =
Addresses.dealership_id
WHERE Addresses.city IS 'Warsaw';
```

6. Bibliografia

- ➔ Slajdy wykładowe WBD sem. 17L
- ➔ Dokumentacja online Oracle
- ➔ W3Schools

Załączniki

Załącznik nr 1



Załącznik nr 2