

# Bazy Danych i Big Data

Oceanarium - Dokumentacja projektu

Michał Jęczmieniowski 325282

Igor Czunikin-Krasowicki 325265

Politechnika Warszawska, Wydział Elektroniki i Technik Informacyjnych

30 Listopada 2023

**Politechnika  
Warszawska**

Prowadzący: dr hab. inż. Marcin Kowalczyk

## Spis treści

<b>1. Zakres i cel projektu</b>	3
1.1. Opis założeń funkcjonalnych projektowanej bazy danych	3
1.2. Użyte oprogramowanie	3
<b>2. Definicja systemu</b>	4
2.1. Perspektywy użytkowników	4
<b>3. Model konceptualny</b>	4
3.1. Definicja zbiorów encji określonych w projekcie (decyzje projektowe)	4
3.2. Ustalenie związków między encjami i ich typów	5
3.3. Określenie atrybutów i ich dziedzin	6
3.4. Dodatkowe reguły integralnościowe (reguły biznesowe)	8
3.5. Klucze kandydujące i główne (decyzje projektowe)	9
3.6. Schemat ER na poziomie konceptualnym	9
3.7. Problem pułapek szczelinowych i wachlarzowych – analiza i przykłady	10
<b>4. Model logiczny</b>	11
4.1. Charakterystyka modelu relacyjnego	11
4.2. Usunięcie właściwości niekompatybilnych z modelem relacyjnym - przykłady	11
4.3. Proces normalizacji – analiza i przykłady	11
4.4. Schemat ER na poziomie modelu logicznego	12
4.5. Więzy integralności	12
4.6. Proces denormalizacji – analiza i przykłady	13
<b>5. Faza fizyczna</b>	14
5.1. Projekt transakcji i weryfikacja ich wykonalności	14
5.2. Strojanie bazy danych – dobór indeksów	14
5.3. Skrypt SQL zakładający bazę danych	16
5.4. Przykłady zapytań i poleceń SQL odnoszących się do bazy danych	24
<b>6. Bibliografia</b>	25

## 1. Zakres i cel projektu

Celem naszego projektu było opracowanie i wdrożenie do życia relacyjnej bazy danych. Aby cel ten można było uznać za wykonany, wymagana jest realizacja trzech podstawowych faz składających się na cały projekt:

- konceptualnej
- relacyjnej
- oraz fizycznej.

Pierwsze dwie fazy dotyczą modelowania ogólnego schematu bazy danych (model konceptualny i powstały na jego podstawie model logiczny). Natomiast trzecia faza dotyczy już komplementarnego wdrożenia implementacji w wybranym przez nasz zespół silniku bazy danych.

### 1.1. Opis założeń funkcjonalnych projektowanej bazy danych

Wykonana przez nas baza danych dotyczy obsługi i zarządzania oceanarium. W bazie, prócz podstawowych informacji na temat oceanarium, zawarliśmy dane opisujące zbiorniki wodne znajdujące się w naszym parku. W każdym z takich zbiorników żyją zwierzęta, które rozróżniamy na podstawie gatunku lub np. rodzaju wody w jakim żyją. Prócz samych zwierząt, w całym oceanarium znajduje się wiele różnych atrakcji, z których mogą korzystać odwiedzający. Mogą oni przebywać na terenie oceanarium, tylko gdy uprzednio, na miejscu bądź on-line, zakupią bilet. Park zatrudnia pracowników na różnego rodzaju stanowiska, wśród których można wyróżnić posadę opiekuna zwierząt, pracownika obsługi oraz przewodnika. Przewodnik może oprowadzać zagraniczne wycieczki, jeśli zna języki obce. Oczywiście każdemu z pracowników przysługuje należne wynagrodzenie. Na terenie oceanarium prowadzone są również sklepy, które dzierżawią teren i oferują własne usługi.

### 1.2. Użyte oprogramowanie

Oprogramowanie, które zostało wykorzystane podczas realizacji projektu:

- Lokalna baza danych Oracle Database w wersji 18c
- Narzędzia do projektowania bazy danych Toad Data Modeler w wersji 7.2
- Narzędzia graficznego do zarządzania bazą danych SQL Developer w wersji 22.2.1
- Oprogramowania Java w wersji 8

## 2. Definicja systemu

Definicja systemu zakłada takie funkcjonalności:

- podgląd informacji o oceanarium,
- dodawanie/modyfikowanie/usuwanie informacji o oceanarium
- podgląd informacji o zbiornikach,
- dodawanie/modyfikowanie/usuwanie informacji o zbiornikach,
- podgląd informacji o zwierzętach,
- dodawanie/modyfikowanie/usuwanie informacji o zwierzętach,
- podgląd informacji o atrakcjach,
- dodawanie/modyfikowanie/usuwanie informacji o atrakcjach,
- podgląd informacji o biletach,
- dodawanie/modyfikowanie/usuwanie informacji o biletach,
- podgląd informacji o pracownikach,
- dodawanie/modyfikowanie/usuwanie informacji o pracownikach,
- podgląd informacji o pracownikach jako opiekuna,
- dodawanie/modyfikowanie/usuwanie informacji o pracownikach jako opiekuna,
- podgląd informacji o pracownikach jako przewodnikach,
- dodawanie/modyfikowanie/usuwanie informacji o pracownikach jako przewodnikach,
- podgląd informacji o językach obcych,
- dodawanie/modyfikowanie/usuwanie informacji o językach obcych,
- podgląd informacji o znajomości języków obcych,
- dodawanie/modyfikowanie/usuwanie informacji o znajomości języków obcych,
- podgląd informacji o wynagrodzeniu pracowników,
- dodawanie/modyfikowanie/usuwanie informacji o wynagrodzeniu pracowników.

### 2.1. Perspektywy użytkowników

Pośród wszystkich użytkowników, którzy mogą posługiwać się naszą bazą danych, możemy wyróżnić następujące perspektywy:

- **właściciel oceanarium** - dostęp i możliwość modyfikacji wszystkich danych w bazie,
- **klient** - dostęp do informacji na temat atrakcji i zwierząt znajdujących się na terenie oceanarium,
- **księgowy** - dostęp i możliwość modyfikacji danych dotyczących pracowników i ich wynagrodzeń,
- **pracownik** - dostęp do informacji na temat wynagrodzenia i możliwość modyfikacji swoich danych kontaktowych.

## 3. Model konceptualny

### 3.1. Definicja zbiorów encji określonych w projekcie (decyzje projektowe)

W naszym projekcie na poziomie konceptualnym wyodrębniliśmy następujące encje:

- **Oceanarium** - encja główna zawierająca informacje ogólne na temat oceanarium,
- **Odwiedzający**- encja zawierająca informacje ogólne na temat odwiedzającego oceanarium,
- **Bilet**- encja zawierająca informacje ogólne na temat biletu do oceanarium,
- **Pracownik**- encja zawierająca informacje ogólne na temat pracownika oceanarium,
- **Opiekun**- encja charakteryzująca pracownika oceanarium jako opiekuna,
- **Przewodnik**- encja charakteryzująca pracownika oceanarium jako przewodnika,
- **Zbiornik**- encja zawierająca informacje ogólne na temat zbiorniku w oceanarium,
- **Atrakcja**- encja zawierająca informacje ogólne na temat atrakcji oceanarium,
- **Sklep**- encja zawierająca informacje ogólne na temat sklepu w oceanarium,
- **Zwierze**- encja zawierająca informacje ogólne na temat zwierzęcia oceanarium.

### 3.2. Ustalenie związków między encjami i ich typów

Nazwa relacji	Krotność relacji	Zasady uczestnictwa	Opis
Oceanarium-Odwiedzający	jeden do wielu	Oceanarium jest obowiązkowe dla odwiedzającego; odwiedzający jest opcjonalny dla oceanarium	Oceanarium może gościć wielu odwiedzających, ale może nie gościć ani jednego. Odwiedzający, aby mógł być nazwany odwiedzającym, musi odwiedzić co najmniej jedno oceanarium.
Oceanarium-Atrakcja	jeden do wielu	Oceanarium jest obowiązkowe dla atrakcji; atrakcja jest opcjonalna dla oceanarium	Oceanarium może posiadać wiele atrakcji, ale może nie posiadać żadnej atrakcji. Atrakcja musi się znajdować w jednym oceanarium.
Oceanarium-Pracownik	jeden do wielu	Oceanarium jest obowiązkowe dla atrakcji; pracownik jest opcjonalny dla oceanarium	Oceanarium może zatrudniać wielu pracowników, ale może nie zatrudniać żadnego pracownika (np. tuż po utworzeniu). Za to pracownik, aby być pracownikiem, musi być zatrudniony w jednym oceanarium.
Oceanarium-Zbiornik	jeden do wielu	Oceanarium jest obowiązkowe dla zbiornika; zbiornik jest opcjonalny dla oceanarium	Oceanarium może posiadać wiele zbiorników, lecz może nie mieć żadnego zbiornika. Zbiornik, aby być zbiornikiem, musi być w jednym oceanarium.
Oceanarium-Sklep	jeden do wielu	Oceanarium jest obowiązkowe dla sklepu; sklep jest opcjonalny dla oceanarium	Oceanarium może posiadać wiele sklepów, lecz może nie mieć żadnego sklepu. Natomiast sklep musi istnieć w jednym oceanarium.
Oceanarium-Zwierze	jeden do wielu	Oceanarium jest obowiązkowe dla zwierzęcia; zwierzę jest opcjonalne dla oceanarium	Oceanarium może posiadać wiele zwierząt, lecz może nie mieć żadnego zwierzęcia. Natomiast zwierzę musi mieszkać w jednym oceanarium.
Sklep-Pracownik	jeden do wielu	Sklep jest opcjonalny dla pracownika; pracownik jest obowiązkowy dla sklepu	Sklep może mieć wielu pracowników oraz musi mieć przynajmniej jednego żeby być sklepem. Natomiast pracownik może być przypisany do maksymalnie jednego sklepu lub żadnego.

Bilet-Atrakcja	wiele do wielu	Bilet jest opcjonalny dla atrakcji; atrakcja jest opcjonalna dla biletu	Z biletem można wejść na wiele atrakcji, ale można nie wejść na żadną atrakcję, jeśli np. atrakcje są w remoncie. Atrakcja może być dostępna na wielu biletów, ale może być niedostępna na żadnym bilecie.
Pracownik-Zbiornik	jeden do wielu	Pracownik jest obowiązkowy dla zbiornika; zbiornik jest opcjonalny dla pracownika	Pracownik może posiadać wiele zbiorników pod swoją opieką, ale może nie posiadać żadnego. Za to zbiornik musi mieć co najmniej jednego pracownika, który się nim zajmuje.
Zbiornik-Zwierze	jeden do wielu	Zbiornik jest obowiązkowy dla zwierzęcia; zwierzę jest opcjonalne dla zbiornika	Zbiornik może posiadać wiele zwierząt, jednak może też nie mieć żadnego zwierzęcia. Zwierzę musi znajdować się w jednym zbiorniku.
Odwiedzający-Bilet	jeden do wielu	Odwiedzający jest obowiązkowy dla biletu; bilet jest opcjonalny dla odwiedzającego	Odwiedzający może posiadać wiele biletów, ale może nie posiadać żadnego biletu. Bilet może być dostępny dla wielu odwiedzających lub żadnego.

### 3.3. Określenie atrybutów i ich dziedzin

Tabela 1. Oceanarium

Atrybut	Typ i dziedzina	Czy obowiązkowy?	Opis
ID_oceanarium	Integer	Tak	Unikatowy identyfikator oceanarium
Nazwa	VarChar(20)	Tak	Nazwa oceanarium
Numer_telefonu	VarChar(15)	Tak	Numer telefonu oceanarium
Godziny_otwarcia	DateTime	Tak	Godziny otwarcia oceanarium
Adres_email	VarChar(20)	Tak	Adres email oceanarium
Data_zalozenia	Date	Tak	Data założenia oceanarium
Adres	VarChar(400)	Tak	Adres oceanarium. Pole segmentowe, które zawiera miasto, ulica, numer budynku i lokalu oraz kod pocztowy.

Tabela 2. **Odwiedzający**

Atrybut	Typ i dziedzina	Czy obowiązkowy?	Opis
ID_odwiedzajacego	Integer	Tak	Unikatowy identyfikator odwiedzającego
Imie	VarChar(20)	Tak	Imię odwiedzającego
Nazwisko	VarChar(30)	Tak	Nazwisko odwiedzającego
Data_urodzenia	Date	Nie	Data urodzenia odwiedzającego
Plec	PlecD(K,M)	Tak	Płeć odwiedzającego
Adres	VarChar(400)	Nie	Adres odwiedzającego. Pole segmentowe, które zawiera miasto, ulica, numer budynku i lokalu oraz kod pocztowy.
Adres_email	VarChar(30)	Tak	Adres email odwiedzającego
Numer_telefonu	VarChar(15)	Nie	Numer telefonu odwiedzającego

Tabela 3. **Bilet**

Atrybut	Typ i dziedzina	Czy obowiązkowy?	Opis
ID_biletu	Integer	Tak	Unikatowy identyfikator biletu
Czy_kupiony_online	Boolean	Tak	Informacja czy bilet został kupiony online
Data_waznosci	Date	Tak	Data ważności biletu
Data_skasowania	Date	Nie	Data skasowania biletu
Cena	Money	Tak	Cena biletu

Tabela 4. **Atrakcja**

Atrybut	Typ i dziedzina	Czy obowiązkowy?	Opis
ID_atrakcji	Integer	Tak	Unikatowy identyfikator atrakcji
Nazwa	VarChar(30)	Tak	Nazwa atrakcji
Opis	VarChar(400)	Tak	Opis atrakcji
Cena	Money	Tak	Cena atrakcji

Tabela 5. **Pracownik**

Atrybut	Typ i dziedzina	Czy obowiązkowy?	Opis
ID_pracownika	Integer	Tak	Unikatowy identyfikator pracownika
Imie	VarChar(20)	Tak	Imię pracownika
Drugie_imie	VarChar(20)	Nie	Drugie imię pracownika
Nazwisko	VarChar(30)	Tak	Nazwisko pracownika
Data_urodzenia	Date	Tak	Data urodzenia pracownika
PESEL	Character(11)	Nie	Numer PESEL pracownika
Plec	PlecD(K,M)	Tak	Płeć pracownika
Adres	VarChar(400)	Tak	Adres pracownika. Pole segmentowe, które zawiera miasto, ulica, numer budynku i lokalu oraz kod pocztowy.
Stanowisko	VarChar(20)	Tak	Stanowisko pracownika
Data_zatrudnienia	Date	Tak	Data zatrudnienia pracownika
Nr_konta	Character(26)	Tak	Numer konta pracownika
Adres_email	VarChar(30)	Tak	Adres email pracownika
Numer_telefonu	VarChar(15)	Tak	Numer telefonu pracownika
Wynagrodzenie	Money	Tak	Wynagrodzenie pracownika

Tabela 6. Zbiornik

Atrybut	Typ i dziedzina	Czy obowiązkowy?	Opis
ID_zbiornika	Integer	Tak	Unikatowy identyfikator zbiornika
Nazwa	VarChar(30)	Tak	Nazwa zbiornika
Pojemnosc	Integer	Tak	Pojemność zbiornika

Tabela 7. Sklep

Atrybut	Typ i dziedzina	Czy obowiązkowy?	Opis
ID_sklepu	Integer	Tak	Unikatowy identyfikator sklepu
Nazwa	VarChar(30)	Tak	Nazwa sklepu

Tabela 8. Opiekun

Atrybut	Typ i dziedzina	Czy obowiązkowy?	Opis
Data_szkolenia	Date	Tak	Data szkolenia opiekuna

Tabela 9. Przewodnik

Atrybut	Typ i dziedzina	Czy obowiązkowy?	Opis
Nr_licencji	Character(8)	Tak	Numer licencji przewodnika
Jezyk	VarChar(100)	Nie	d

Tabela 10. Zwierzę

Atrybut	Typ i dziedzina	Czy obowiązkowy?	Opis
ID_zwierzecia	Integer	Tak	Unikatowy identyfikator zwierzęcia
Gatunek	VarChar(30)	Tak	Gatunek zwierzęcia
Imie	VarChar(20)	Nie	Imię zwierzęcia
Typ_wody	Typ_wodyD(SLON, SLOD)	Tak	Typ wody, w którym żyje zwierzę
Data_urodzenia	Date	Nie	Data urodzenia zwierzęcia

### 3.4. Dodatkowe reguły integralnościowe (reguły biznesowe)

W ramach dodatkowych reguł integralnościowych zakładamy że:

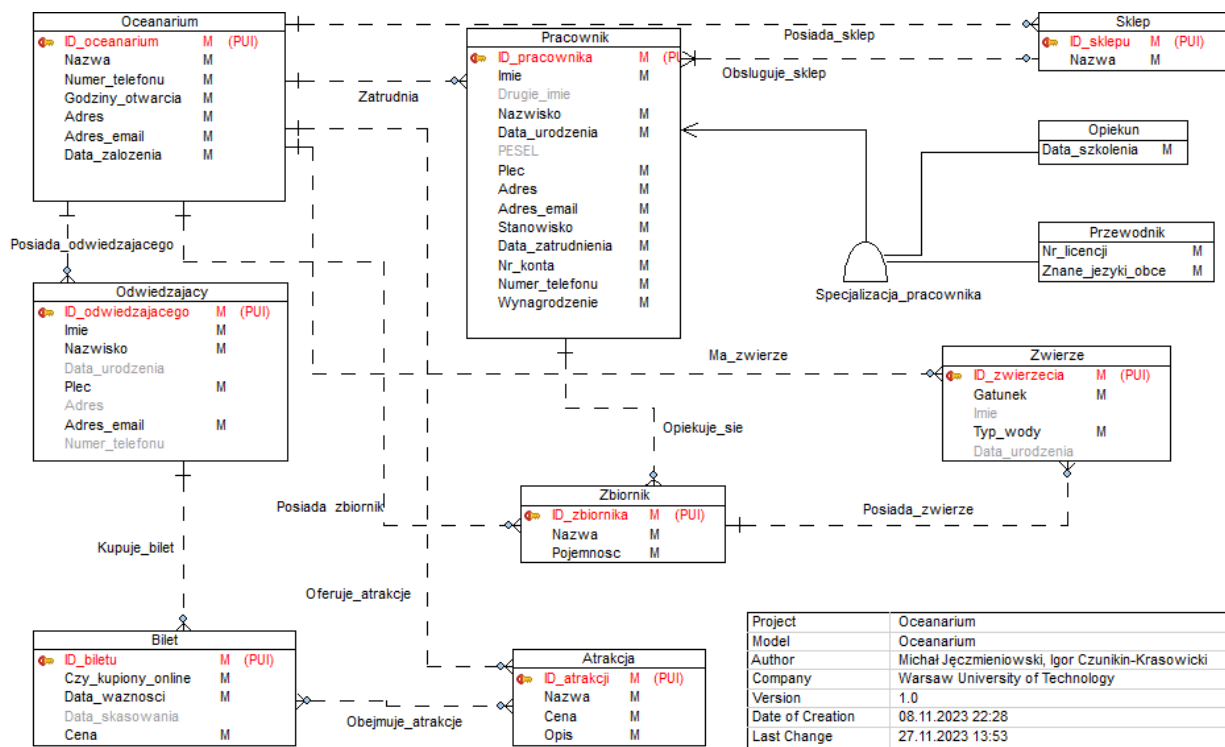
- Zbiornik nigdy nie powinien być kojarzony z atrakcją i vice versa. Wszyscy odwiedzający mogą podziwiać zwierzęta w zbiornikach, natomiast atrakcje to opcjonalne, dodatkowo płatne punkty.
- Bilety kupione stacjonarnie wymagają podania przez odwiedzającego tylko wymaganych informacji (Imię, nazwisko, e-mail), natomiast bilety zakupione online wymagają podania więcej danych (Data urodzenia, adres wymagany do rozliczenia i numer telefonu)



### 3.5. Klucze kandydujące i główne (decyzje projektowe)

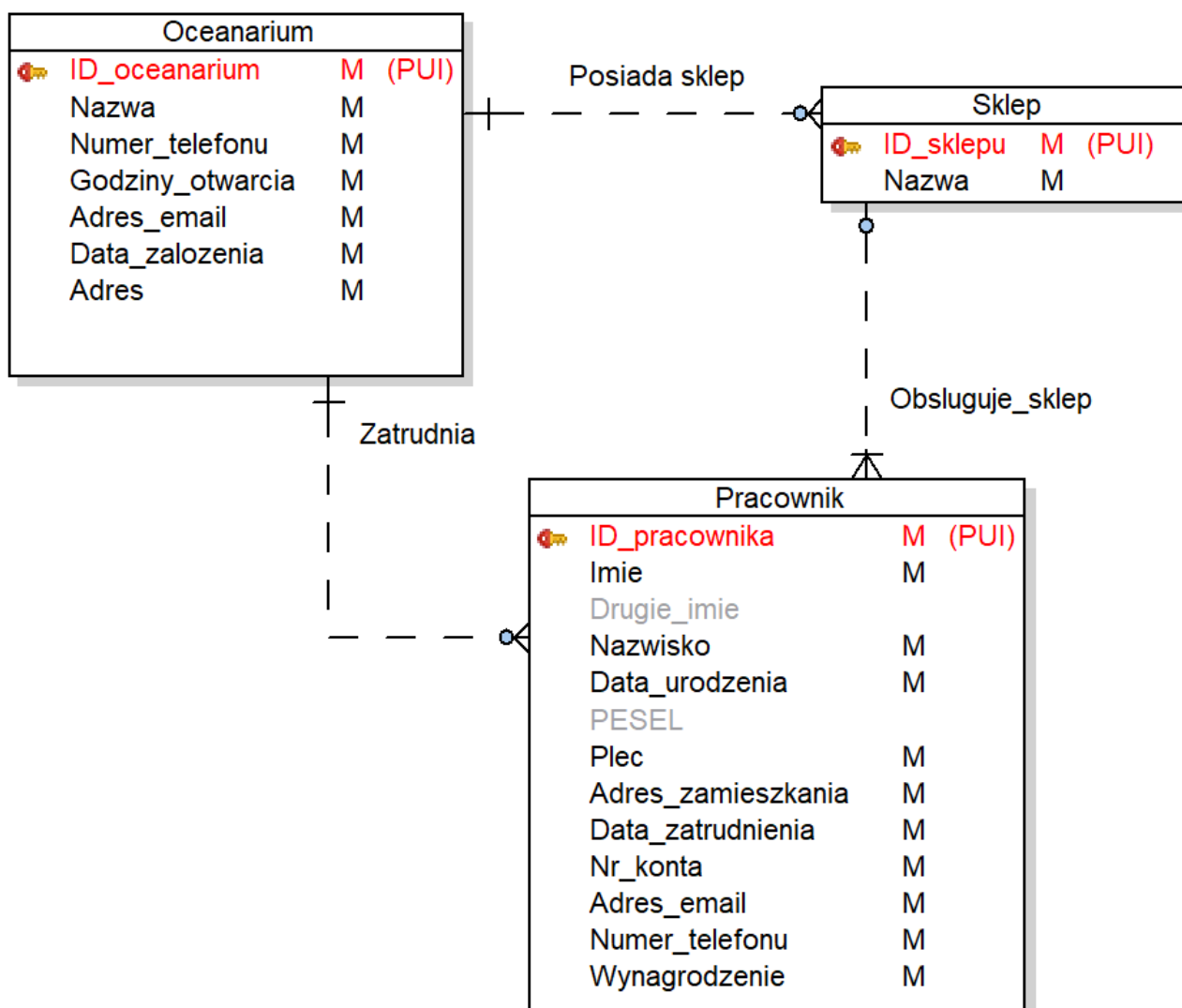
Nazwa encji	Klucz główny	Klucz kandydujący
Oceanarium	ID_oceanarium	Nazwa, Numer_telefonu, Adres_email
Odwiedzający	ID_odwiedzajacego	Adres_email, Numer_telefonu
Bilet	ID_biletu	-
Pracownik	ID_pracownika	PESEL, Adres_email, Numer_telefonu
Opiekun	ID_pracownika	jw.
Przewodnik	ID_pracownika	jw. + Nr_licencji
Zbiornik	ID_zbiornika	Nazwa
Atrakcja	ID_atrakcji	Nazwa
Sklep	ID_sklepu	Nazwa
Zwierze	ID_zwierzecia	-

### 3.6. Schemat ER na poziomie konceptualnym



### 3.7. Problem pułapek szczelinowych i wachlarzowych – analiza i przykłady

Poniżej przedstawiamy przykład rozwiązanych potencjalnych pułapek szczelinowej i wachlarzowej.



**Pułapka szczelinowa** - Gdyby nie istniała relacja **Zatrudnia** wiedzielibyśmy które sklepy należą do konkretnego oceanarium oraz że sklepy posiadają obsługujących je pracowników, natomiast przez brak powiązania **Pracownik-Oceanarium** model zakłada że każdy pracownik jest pracownikiem sklepu co jest oczywiście nieprawdą dla np. opiekunów, przewodników, księgowych itd.

**Pułapka wachlarzowa** - W przypadku w którym nie istniałaby relacja **Obsluguje sklep**, wiedzielibyśmy którzy pracownicy oraz które sklepy należą do danego oceanarium, natomiast przez brak powiązania **Pracownik-Sklep** istniałby wachlarz możliwości co do tego którzy pracownicy obsługują które sklepy i w jakich ilościach.

## 4. Model logiczny

### 4.1. Charakterystyka modelu relacyjnego

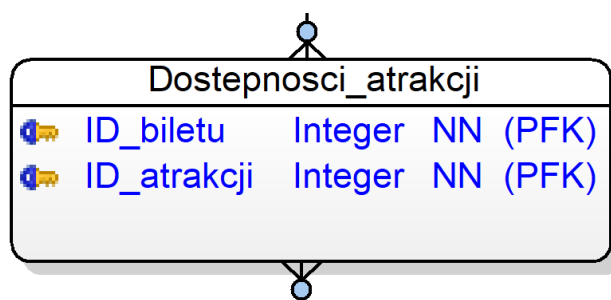
Po weryfikacji naszego schematu ER poprzez funkcję **Verify Model** w programie Toad Data Modeler, zakończonej sukcesem, zakończyliśmy etap projektowania go na poziomie konceptualnym. Tym samym przeszliśmy do etapu, w którym skonwertowaliśmy nasz projekt na poziom logiczny, również poprzez program Toad Data Modeler. Podczas takiej konwersji:

- każda z relacji „wielu do wielu” zostaje zastąpiona dwiema relacjami „jeden do wielu” oraz tabelą łączącą dwie nowo powstałe relacje.
- tabele otrzymują nowe dodatkowe atrybuty pod postacią kluczy obcych (oznaczone są one na zielono)
- niektóre typy danych występujące w modelu konceptualnym zostają przekształcone w procesie konwersji na takie, jakich używa wybrany przez nas silnik bazy danych (np. Money),
- encje dziedziczące zostają uzupełnione o klucz główny encji nadrzędnej (oznaczone na niebiesko).

### 4.2. Usunięcie właściwości niekompatybilnych z modelem relacyjnym - przykłady

Podczas konwersji naszego modelu konceptualnego w model logiczny, narzędzie automatycznie wygenerowało tabelę łączącą, w miejscu występowania relacji „wiele do wielu”. Z racji na automatyczne i mało czytelne nazwanie tej tabeli, zmieniliśmy jej nazwę.

Jedyną nowo powstałą tabelą jest ta pomiędzy „Bilety”, a „Atrakcje”. Zmieniliśmy jej nazwę na „Dostępności\_atrakcji” jej celem jest przechowanie informacji, o tym jaki bilet korzystał z której atrakcji podczas pobytu na terenie oceanarium.



Widzimy że powyższa tabela posiada nazwę w liczbie mnogiej. W celu odróżnienia encji (tak nazywa się tabele powstałe na poziomie projektowania konceptualnego) od relacji (tak nazywa się tabele powstałe na poziomie projektowania logicznego), nazwę każdej tabeli powinno się zamienić na liczbę mnogą. Każde następne tabele tworzone na tym etapie również powinny mieć w nazwie liczbę mnogą.

### 4.3. Proces normalizacji – analiza i przykłady

Zadaniem procesu normalizacji jest usunięcie powtarzających się danych w całej bazie, i co przy tym równie ważnym, organizacja ich w jednym wybranym miejscu. Pomoże to w większym zachowaniu bezpieczeństwa jak i łatwiejszej obsłudze bazy, lepszemu dostępowi do informacji oraz prostszej możliwości utrzymania bazy danych jako spójnej całości.

Podczas procesu normalizacji sprawdzamy, czy baza danych spełnia wymogi trzeciej postaci normalnej (3PN). Najpierw jednak wymagana jest pierwsza postać normalna (1PN), według której każda wartość atrybutu w każdej danej relacji ma być wartością atomową i fakt, aby w każdej relacji nie było powtarzających się grup. W naszym modelu konceptualnym istniały 3 atrybuty, które musieliśmy zmienić, aby spełniać 1PN:

- atrybut „Adres”, który występował w relacjach „Oceanarium”, „Odwiedzający” i „Pracownik”, był polem segmentowym, czyli zawierał więcej niż jeden typ wartości (np. kraj, miasto, ulicę); problem ten rozwiązaliśmy tworząc nową relację o tej nazwie, służącą do przechowywania danych adresowych oddzielnie, a dopiero tą nowo powstałą relację połączyliśmy z trzema wyżej wymienionymi relacjami,
- atrybut „Język” występujący w relacji „Przewodnik” był polem wielowartościowym co oznacza, że możliwe było przechowanie kilka takich samych typu wartości (np. przewodnik może znać wiele języków obcych) czyli przechowywanie wielu wartości, oraz polem segmentowym, przechowującym informacje o języku oraz stopniu

jego znajomości; podobnie jak wcześniej, utworzyliśmy nową relację do zawarcia w niej języków obcych oraz tabelę łączącą relacje „Języki” i „Przewodnicy” z atrybutem „Poziom\_jezyka” z racji tego, że powstała tam kolejna relacja „wiele do wielu” (przewodnik może znać wiele języków, języki mogą być znane przez wielu przewodników).

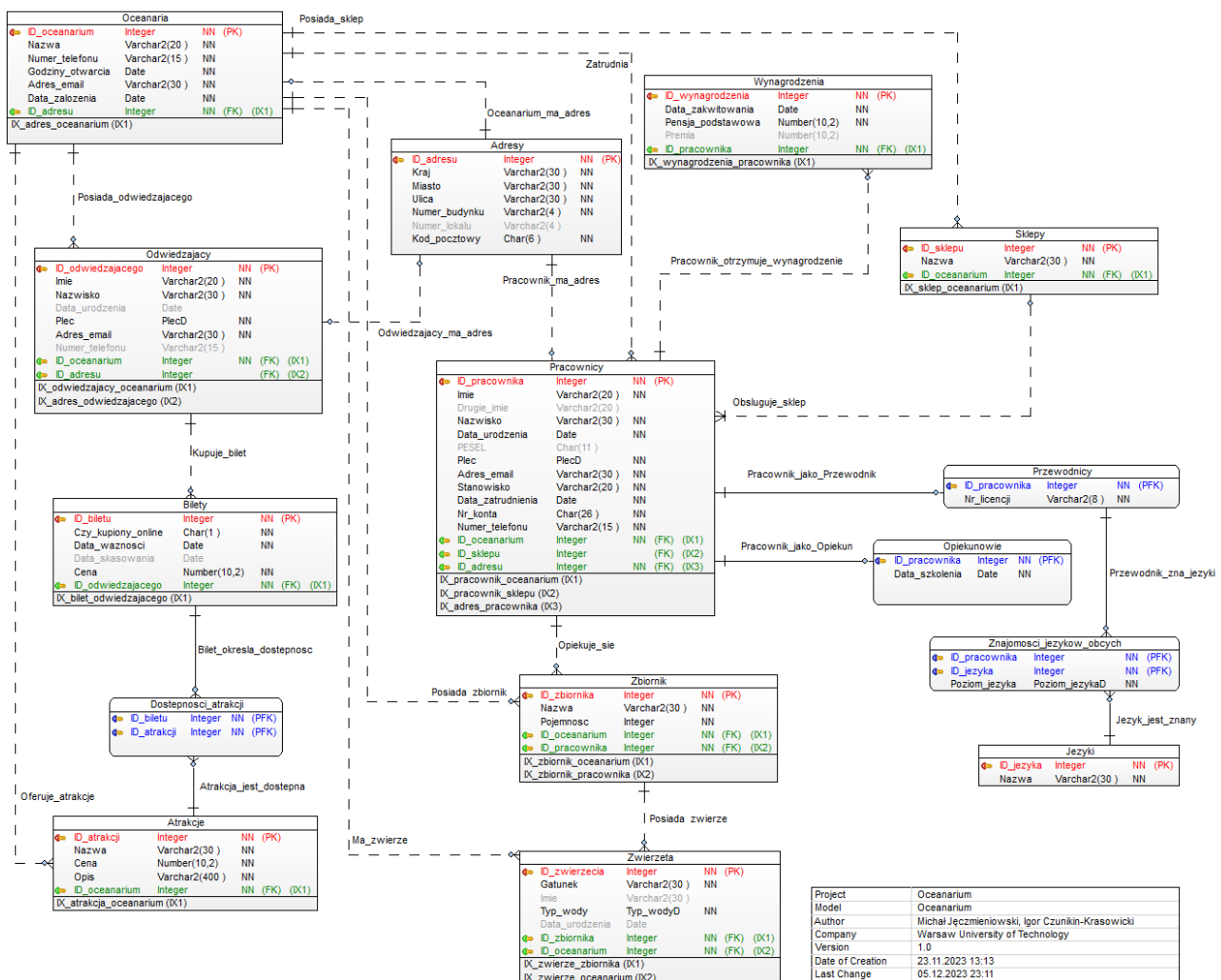
- atrybut „Wynagrodzenie” występujący w relacji „Pracownik” był polem segmentowym, ponieważ zawiera informacje o pensji podstawowej, premii czy dacie zakwitowania; utworzyliśmy oddzielną relację z takim atrybutami i połączyliśmy ją z relacją „Pracownicy”.

Drugą postać normalną (2PN) można osiągnąć jeśli relacja jest w 1PN, każdy atrybut nie wchodzący w skład żadnego klucza potencjalnego ma być w pełni funkcyjnie zależny od wszystkich innych kluczy potencjalnych relacji oraz fakt, że każdy atrybut relacji niewchodzący w skład klucza zależał w pełni od całego klucza a nie tylko jego części.

W naszej bazie danych każdy klucz jest kluczem prostym, kryterium to zostało spełnione.

Trzecia postać normalna (3PN) wymaga realizacji 2PN, a także żadna informacja w kolumnie, która nie jest kluczem podstawowym, nie może zależeć od niczego innego, jak tylko od klucza podstawowego. Ten stan spójności również udało nam się osiągnąć bez większych zmian w projekcie.

#### 4.4. Schemat ER na poziomie modelu logicznego



#### 4.5. Więzy integralności

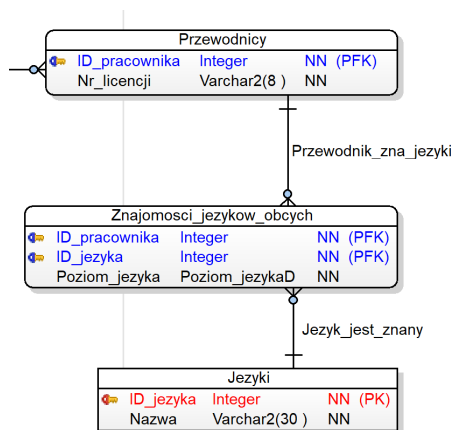
Realizacja warunków integralności danych oznacza zagwarantowanie zgodności między treścią pól rekordów a ich typami w bazie danych. Jeśli na etapie projektowania określono, że atrybut „Data\_zatrudnienia” ma typ

"Date", to pole tego rekordu przechowujące datę zatrudnienia pracownika musi być datą, a nie na przykład liczbą zmiennoprzecinkową. W projektowanej bazie danych większość atrybutów to obowiązkowe pola określone przez konkretny typ, co minimalizuje sytuacje, w których dopuszcza się wartości NULL dla danego atrybutu w trakcie użytkowania systemu.

Dodatkowo, każde pole kluczowe w danej relacji powinno być unikalne, co zostało osiągnięte zgodnie z zasadami normalizacji, jak opisano w ostatnim akapicie tego rozdziału.

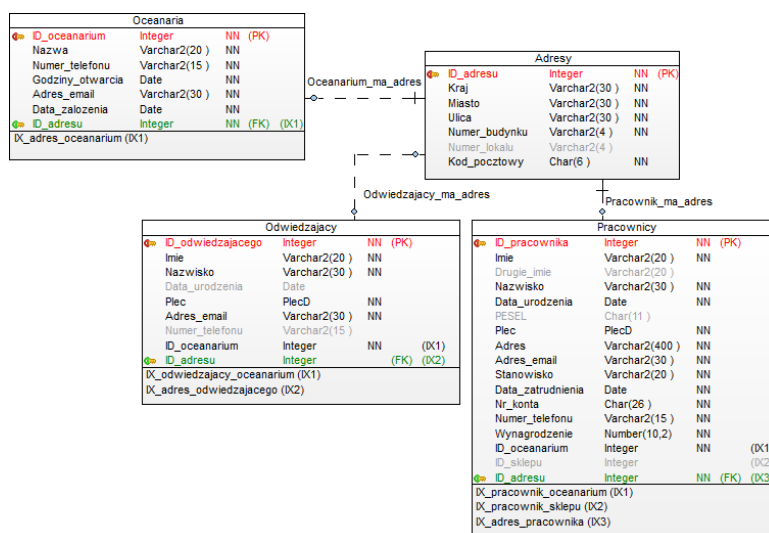
#### 4.6. Proces denormalizacji – analiza i przykłady

Po znormalizowaniu bazy danych często rozważa się zastosowanie odwrotnej operacji (denormalizacji), polegającej na połączeniu niektórych znormalizowanych tabel, z myślą o przyspieszeniu dostępu do pewnych danych. Możliwy przykład użycia denormalizacji:



Jeśli chcielibyśmy często skorzystać z informacji o znanych językach, jakie zna Przewodnik, moglibyśmy poddać procesowi denormalizacji powyższe tabele poprzez przeniesienie atrybutów z tabeli **Jezyki** do tabeli **Znajomosci\_jezykow\_obcych**, zrezygnowaliśmy z połączenia tych tabel aby uniknąć redundancji danych ponieważ nasz projekt nie zakłada częstej potrzeby pytania o znane języki przez przewodników.

Z kolei denormalizacja poniższych relacji mogłaby doprowadzić do poważnych konsekwencji:



Usunięcie relacji Adresy i wprowadzenie wszystkich ich atrybutów do relacji Pracownicy, Odwiedzajacy oraz Oceanaria spowodowałoby olbrzymią redundancję danych w naszej bazie. Relacje te mają zbyt dużo związków z innymi relacjami, żeby denormalizacja była opłacalna i w większość baz danych działałaby przez to z opóźnieniem, co nie jest celem denormalizacji

## 5. Faza fizyczna

### 5.1. Projekt transakcji i weryfikacja ich wykonalności

Oto tabela:

Transakcja	Weryfikacja wykonalności	Potrzebne relacje
Podgląd informacji o oceanarium	Wykonalne	Oceanaria, Adresy
Dodawanie/modyfikowanie/usuwanie informacji o oceanarium	Wykonalne	Oceanaria, Adresy
Podgląd informacji o zbiornikach	Wykonalne	Oceanaria, Zbiorniki
Dodawanie/modyfikowanie/usuwanie informacji o zbiornikach	Wykonalne	Oceanaria, Zbiorniki
Podgląd informacji o zwierzętach	Wykonalne	Oceanaria, Zwierzeta
Dodawanie/modyfikowanie/usuwanie informacji o zwierzętach	Wykonalne	Oceanaria, Zwierzeta
Podgląd informacji o atrakcjach	Wykonalne	Oceanaria, Atrakcje
Dodawanie/modyfikowanie/usuwanie informacji o atrakcjach	Wykonalne	Oceanaria, Atrakcje
Podgląd informacji o biletach	Wykonalne	Oceanaria, Bilety, Odwiedzający
Dodawanie/modyfikowanie/usuwanie informacji o biletach	Wykonalne	Oceanaria, Bilety, Odwiedzający
Podgląd informacji o pracownikach	Wykonalne	Oceanaria, Pracownicy, Adresy, Wynagrodzenia
Dodawanie/modyfikowanie/usuwanie informacji o pracownikach	Wykonalne	Oceanaria, Pracownicy, Adresy, Wynagrodzenia
Podgląd informacji o pracownikach jako opiekuna	Wykonalne	Pracownicy, Opiekunowie
Dodawanie/modyfikowanie/usuwanie informacji o pracownikach jako opiekuna	Wykonalne	Pracownicy, Opiekunowie
Podgląd informacji o pracownikach jako przewodników	Wykonalne	Pracownicy, Przewodnicy
Dodawanie/modyfikowanie/usuwanie informacji o pracownikach jako przewodników	Wykonalne	Pracownicy, Przewodnicy
Podgląd informacji o językach obcych	Wykonalne	Opiekunowie, Znajomości_językow_obcych, Języki
Dodawanie/modyfikowanie/usuwanie informacji o językach obcych	Wykonalne	Opiekunowie, Znajomości_językow_obcych, Języki
Podgląd informacji o znajomości języków obcych	Wykonalne	Znajomości_językow_obcych, Języki
Dodawanie/modyfikowanie/usuwanie informacji o znajomości języków obcych	Wykonalne	Znajomości_języki_obcych, Języki
Podgląd informacji o wynagrodzeniu pracowników	Wykonalne	Pracownicy, Wynagrodzenie
Dodawanie/modyfikowanie/usuwanie informacji o wynagrodzeniu	Wykonalne	Pracownicy, Wynagrodzenie

### 5.2. Strojanie bazy danych – dobór indeksów

Do strojenia bazy danych będą nam służyć wygenerowane indeksy, automatycznie dodawane przy okazji tworzenia relacji między naszymi tabelami:

- **IX\_adres\_oceanarium** - wyszukiwanie adresu oceanariów

```
1 CREATE INDEX IX_adres_oceanarium ON Oceanaria (ID_adresu)
```

- **IX\_odwiedzajacy\_oceanarium** - wyszukiwanie odwiedzających oceanariów

```
1 CREATE INDEX IX_odwiedzajacy_oceanarium ON Odwiedzajacy (ID_oceanarium)
```

- **IX\_adres\_odwiedzajacego** - wyszukiwanie adresów odwiedzających

```
1 CREATE INDEX IX_adres_odwiedzajacego ON Odwiedzajacy (ID_adresu)
```

- **IX\_bilet\_odwiedzajacego** - wyszukiwanie biletów odwiedzających

```
1 CREATE INDEX IX_bilet_odwiedzajacego ON Bilety (ID_odwiedzajacego)
```

- **IX\_atrakcja\_oceanarium** - wyszukiwanie atrakcji oceanariów

```
1 CREATE INDEX IX_atrakcja_oceanarium ON Atrakcje (ID_oceanarium)
```

- **IX\_pracownik\_oceanarium** - wyszukiwanie pracowników oceanariów

```
1 CREATE INDEX IX_pracownik_oceanarium ON Pracownicy (ID_oceanarium)
```

- **IX\_pracownik\_sklepu** - wyszukiwanie pracowników sklepów

```
1 CREATE INDEX IX_pracownik_sklepu ON Pracownicy (ID_sklepu)
```

- **IX\_adres\_pracownika** - wyszukiwanie adresów pracowników

```
1 CREATE INDEX IX_adres_pracownika ON Pracownicy (ID_adresu)
```

- **IX\_zbiornik\_oceanarium** - wyszukiwanie zbiorników oceanariów

```
1 CREATE INDEX IX_zbiornik_oceanarium ON Zbiornik (ID_oceanarium)
```

- **IX\_zbiornik\_pracownika** - wyszukiwanie zbiorników pracowników

```
1 CREATE INDEX IX_zbiornik_pracownika ON Zbiornik (ID_pracownika)
```

- **IX\_zwierze\_zbiornika** - wyszukiwanie zwierząt zbiorników

```
1 CREATE INDEX IX_zwierze_zbiornika ON Zwierzeta (ID_zbiornika)
```

- **IX\_zwierze\_oceanarium** - wyszukiwanie zwierząt oceanariów

```
1 CREATE INDEX IX_zwierze_oceanarium ON Zwierzeta (ID_oceanarium)
```

- **IX\_sklep\_oceanarium** - wyszukiwanie sklepów oceanariów

```
1 CREATE INDEX IX_sklep_oceanarium ON Sklepy (ID_oceanarium)
```

### 5.3. Skrypt SQL zakładający bazę danych

```
1  /*
2  Created: 23.11.2023
3  Modified: 27.11.2023
4  Project: Oceanarium
5  Model: Oceanarium
6  Company: Warsaw University of Technology
7  Author: Michal Jeczmiński, Igor Czunikin-Krasowicki
8  Version: 1.0
9  Database: Oracle 18c
10 */
11
12
13 -- Create tables section -----
14
15 -- Table Oceanaria
16
17 CREATE TABLE Oceanaria(
18     ID_oceanarium Integer NOT NULL,
19     Nazwa Varchar2(20 ) NOT NULL,
20     Numer_telefonu Varchar2(15 ) NOT NULL,
21     Godziny_otwarcia Varchar2(5) NOT NULL,
22     Adres_email Varchar2(30 ) NOT NULL,
23     Data_zalozenia Date NOT NULL,
24     ID_adresu Integer NOT NULL
25 )
26 /
27
28 -- Create indexes for table Oceanaria
29
30 CREATE INDEX IX_adres_oceanarium ON Oceanaria (ID_adresu)
31 /
32
33 -- Add keys for table Oceanaria
34
35 ALTER TABLE Oceanaria ADD CONSTRAINT Unique_Identifier1 PRIMARY KEY (ID_oceanarium)
36 /
37
38 -- Table Pracownicy
39
40 CREATE TABLE Pracownicy(
41     ID_pracownika Integer NOT NULL,
42     Imie Varchar2(20 ) NOT NULL,
43     Drugie_imie Varchar2(20 ),
44     Nazwisko Varchar2(30 ) NOT NULL,
45     Data_urodzenia Date NOT NULL,
46     PESEL Char(11 ),
47     Plec Char(1 ) NOT NULL
48         CHECK (Plec IN ('K','M')),
49     Adres_email Varchar2(30 ) NOT NULL,
50     Stanowisko Varchar2(20 ) NOT NULL,
51     Data_zatrudnienia Date NOT NULL,
52     Nr_konta Char(26 ) NOT NULL,
```



```

53     Numer_telefonu Varchar2(15 ) NOT NULL,
54     ID_oceanarium Integer NOT NULL,
55     ID_sklepu Integer,
56     ID_adresu Integer NOT NULL
57 )
58 /
59
60 -- Create indexes for table Pracownicy
61
62 CREATE INDEX IX_pracownik_oceanarium ON Pracownicy (ID_oceanarium)
63 /
64
65 CREATE INDEX IX_pracownik_sklepu ON Pracownicy (ID_sklepu)
66 /
67
68 CREATE INDEX IX_adres_pracownika ON Pracownicy (ID_adresu)
69 /
70
71 -- Add keys for table Pracownicy
72
73 ALTER TABLE Pracownicy ADD CONSTRAINT Unique_Identifier2 PRIMARY KEY (ID_pracownika)
74 /
75
76 -- Table Odwiedzajacy
77
78 CREATE TABLE Odwiedzajacy(
79     ID_odwiedzajacego Integer NOT NULL,
80     Imie Varchar2(20 ) NOT NULL,
81     Nazwisko Varchar2(30 ) NOT NULL,
82     Data_urodzenia Date,
83     Plec Char(1 ) NOT NULL
84         CHECK (Plec IN ('K','M')),
85     Adres_email Varchar2(30 ) NOT NULL,
86     Numer_telefonu Varchar2(15 ),
87     ID_oceanarium Integer NOT NULL,
88     ID_adresu Integer
89 )
90 /
91
92 -- Create indexes for table Odwiedzajacy
93
94 CREATE INDEX IX_odwiedzajacy_oceanarium ON Odwiedzajacy (ID_oceanarium)
95 /
96
97 CREATE INDEX IX_adres_odwiedzajacego ON Odwiedzajacy (ID_adresu)
98 /
99
100 -- Add keys for table Odwiedzajacy
101
102 ALTER TABLE Odwiedzajacy ADD CONSTRAINT Unique_Identifier3
103 PRIMARY KEY (ID_odwiedzajacego)
104 /
105
106 -- Table Zwierzeta
107
108 CREATE TABLE Zwierzeta(
109     ID_zwierzecia Integer NOT NULL,
110     Gatunek Varchar2(30 ) NOT NULL,
111     Imie Varchar2(30 ),
112     Typ_wody Char(4 ) NOT NULL
113         CHECK (TYP_WODY IN ('SLOD','SLON')),

```

```

114     Data_urodzenia Date,
115     ID_zbiornika Integer NOT NULL,
116     ID_oceanarium Integer NOT NULL
117 )
118 /
119
120 -- Create indexes for table Zwierzeta
121
122 CREATE INDEX IX_zwierzze_zbiornika ON Zwierzeta (ID_zbiornika)
123 /
124
125 CREATE INDEX IX_zwierzze_oceanarium ON Zwierzeta (ID_oceanarium)
126 /
127
128 -- Add keys for table Zwierzeta
129
130 ALTER TABLE Zwierzeta ADD CONSTRAINT Unique_Identifier4 PRIMARY KEY (ID_zwierzecia)
131 /
132
133 -- Table Zbiornik
134
135 CREATE TABLE Zbiornik(
136     ID_zbiornika Integer NOT NULL,
137     Nazwa Varchar2(30 ) NOT NULL,
138     Pojemnosc Integer NOT NULL,
139     ID_oceanarium Integer NOT NULL,
140     ID_pracownika Integer NOT NULL
141 )
142 /
143
144 -- Create indexes for table Zbiornik
145
146 CREATE INDEX IX_zbiornik_oceanarium ON Zbiornik (ID_oceanarium)
147 /
148
149 CREATE INDEX IX_zbiornik_pracownika ON Zbiornik (ID_pracownika)
150 /
151
152 -- Add keys for table Zbiornik
153
154 ALTER TABLE Zbiornik ADD CONSTRAINT Unique_Identifier5 PRIMARY KEY (ID_zbiornika)
155 /
156
157 -- Table Sklepy
158
159 CREATE TABLE Sklepy(
160     ID_sklepu Integer NOT NULL,
161     Nazwa Varchar2(30 ) NOT NULL,
162     ID_oceanarium Integer NOT NULL
163 )
164 /
165
166 -- Create indexes for table Sklepy
167
168 CREATE INDEX IX_sklep_oceanarium ON Sklepy (ID_oceanarium)
169 /
170
171 -- Add keys for table Sklepy
172
173 ALTER TABLE Sklepy ADD CONSTRAINT Unique_Identifier6 PRIMARY KEY (ID_sklepu)
174 /

```

```

175
176 -- Table Bilety
177
178 CREATE TABLE Bilety(
179     ID_biletu Integer NOT NULL,
180     Czy_kupiony_online Char(1 ) NOT NULL,
181     Data_waznosci Date NOT NULL,
182     Data_skasowania Date,
183     Cena Number(10,2) NOT NULL,
184     ID_odwiedzajacego Integer NOT NULL
185 )
186 /
187
188 -- Create indexes for table Bilety
189
190 CREATE INDEX IX_bilet_odwiedzajacego ON Bilety (ID_odwiedzajacego)
191 /
192
193 -- Add keys for table Bilety
194
195 ALTER TABLE Bilety ADD CONSTRAINT Unique_Identifier7 PRIMARY KEY (ID_biletu)
196 /
197
198 -- Table Atrakcje
199
200 CREATE TABLE Atrakcje(
201     ID_atrakcji Integer NOT NULL,
202     Nazwa Varchar2(30 ) NOT NULL,
203     Cena Number(10,2) NOT NULL,
204     Opis Varchar2(400 ) NOT NULL,
205     ID_oceanarium Integer NOT NULL
206 )
207 /
208
209 -- Create indexes for table Atrakcje
210
211 CREATE INDEX IX_atrakcja_oceanarium ON Atrakcje (ID_oceanarium)
212 /
213
214 -- Add keys for table Atrakcje
215
216 ALTER TABLE Atrakcje ADD CONSTRAINT Unique_Identifier8 PRIMARY KEY (ID_atrakcji)
217 /
218
219 -- Table Opiekunowie
220
221 CREATE TABLE Opiekunowie(
222     ID_pracownika Integer NOT NULL,
223     Data_szkolenia Date NOT NULL
224 )
225 /
226
227 -- Add keys for table Opiekunowie
228
229 ALTER TABLE Opiekunowie ADD CONSTRAINT Unique_Identifier9 PRIMARY KEY (ID_pracownika)
230 /
231
232 -- Table Przewodnicy
233
234 CREATE TABLE Przewodnicy(
235     ID_pracownika Integer NOT NULL,

```

```

236     Nr_licencji Varchar2(8 ) NOT NULL
237 )
238 /
239
240 -- Add keys for table Przewodnicy
241
242 ALTER TABLE Przewodnicy ADD CONSTRAINT Unique_Identifier10 PRIMARY KEY (ID_pracownika)
243 /
244
245 -- Table Dostepnosci_atrakcji
246
247 CREATE TABLE Dostepnosci_atrakcji(
248     ID_biletu Integer NOT NULL,
249     ID_atrakcji Integer NOT NULL
250 )
251 /
252
253 -- Table Adresy
254
255 CREATE TABLE Adresy(
256     ID_adresu Integer NOT NULL,
257     Kraj Varchar2(30 ) NOT NULL,
258     Miasto Varchar2(30 ) NOT NULL,
259     Ulica Varchar2(30 ) NOT NULL,
260     Numer_budynku Varchar2(4 ) NOT NULL,
261     Numer_lokalu Varchar2(4 ),
262     Kod_pocztowy Char(6 ) NOT NULL
263 )
264 /
265
266 -- Add keys for table Adresy
267
268 ALTER TABLE Adresy ADD CONSTRAINT PK_Adresy PRIMARY KEY (ID_adresu)
269 /
270
271 -- Table and Columns comments section
272
273 COMMENT ON COLUMN Adresy.ID_adresu IS 'Unikalny identyfikator adresu'
274 /
275 COMMENT ON COLUMN Adresy.Kraj IS 'Kraj w adresie'
276 /
277 COMMENT ON COLUMN Adresy.Miasto IS 'Miasto w adresie'
278 /
279 COMMENT ON COLUMN Adresy.Ulica IS 'Ulica w adresie'
280 /
281 COMMENT ON COLUMN Adresy.Numer_budynku IS 'Numer budynku w adresie'
282 /
283 COMMENT ON COLUMN Adresy.Numer_lokalu IS 'Numer lokalu w adresie'
284 /
285 COMMENT ON COLUMN Adresy.Kod_pocztowy IS 'Kod pocztowy w adresie'
286 /
287
288 -- Table Znajomosci_jezykow_obcych
289
290 CREATE TABLE Znajomosci_jezykow_obcych(
291     ID_pracownika Integer NOT NULL,
292     ID_jezyka Integer NOT NULL,
293     Poziom_jezyka Char(20 ) NOT NULL
294         CHECK (POZIOM_JEZYKA IN ('A1','A2','B1','B2','C1','C2'))
295 )
296 /

```

```

297
298 -- Add keys for table Znajomosci_jezykow_obcych
299
300 ALTER TABLE Znajomosci_jezykow_obcych ADD CONSTRAINT PK_Znajomosci_jezykow_obcych
301 PRIMARY KEY (ID_pracownika, ID_jezyka)
302 /
303
304 -- Table and Columns comments section
305
306 COMMENT ON COLUMN Znajomosci_jezykow_obcych.Poziom_jezyka IS 'Poziom jezyka'
307 /
308
309 -- Table Jezyki
310
311 CREATE TABLE Jezyki(
312     ID_jezyka Integer NOT NULL,
313     Nazwa Varchar2(30 ) NOT NULL
314 )
315 /
316
317 -- Add keys for table Jezyki
318
319 ALTER TABLE Jezyki ADD CONSTRAINT PK_Jezyki PRIMARY KEY (ID_jezyka)
320 /
321
322 -- Table and Columns comments section
323
324 COMMENT ON COLUMN Jezyki.ID_jezyka IS 'Unikalny identyfikator jezyka'
325 /
326 COMMENT ON COLUMN Jezyki.Nazwa IS 'Nazwa jezyka'
327 /
328
329 -- Table Wynagrodzenia
330
331 CREATE TABLE Wynagrodzenia(
332     ID_wynagrodzenia Integer NOT NULL,
333     Data_zakwitowania Date NOT NULL,
334     Pensja_podstawowa Number(10,2) NOT NULL,
335     Premia Number(10,2),
336     ID_pracownika Integer NOT NULL
337 )
338 /
339
340 -- Create indexes for table Wynagrodzenia
341
342 CREATE INDEX IX_wynagrodzenia_pracownika ON Wynagrodzenia (ID_pracownika)
343 /
344
345 -- Add keys for table Wynagrodzenia
346
347 ALTER TABLE Wynagrodzenia ADD CONSTRAINT PK_Wynagrodzenia
348 PRIMARY KEY (ID_wynagrodzenia)
349 /
350
351 -- Table and Columns comments section
352
353 COMMENT ON COLUMN Wynagrodzenia.ID_wynagrodzenia IS
354 'Unikalny identyfikator wynagrodzenia'
355 /
356 COMMENT ON COLUMN Wynagrodzenia.Data_zakwitowania IS
357 'Data zakwitowania wynagrodzenia'

```

```

358 /
359 COMMENT ON COLUMN Wynagrodzenia.Pensja_podstawowa IS 'Pensja podstawowa pracownika'
360 /
361 COMMENT ON COLUMN Wynagrodzenia.Premia IS 'Premia do wynagrodzenia'
362 /
363
364
365 -- Create foreign keys (relationships) section -----
366
367 ALTER TABLE Pracownicy ADD CONSTRAINT Zatrudnia FOREIGN KEY (ID_oceanarium)
368 REFERENCES Oceanaria (ID_oceanarium)
369 /
370
371
372
373 ALTER TABLE Odwiedzajacy ADD CONSTRAINT Posiada_odwiedzajacego FOREIGN KEY
374 (ID_oceanarium) REFERENCES Oceanaria (ID_oceanarium)
375 /
376
377
378
379 ALTER TABLE Zbiornik ADD CONSTRAINT Posiada_zbiornik FOREIGN KEY (ID_oceanarium)
380 REFERENCES Oceanaria (ID_oceanarium)
381 /
382
383
384
385 ALTER TABLE Zwierzeta ADD CONSTRAINT Posiada_zwierze FOREIGN KEY (ID_zbiornika)
386 REFERENCES Zbiornik (ID_zbiornika)
387 /
388
389
390
391 ALTER TABLE Sklepy ADD CONSTRAINT Posiada_sklep FOREIGN KEY (ID_oceanarium)
392 REFERENCES Oceanaria (ID_oceanarium)
393 /
394
395
396
397 ALTER TABLE Bilety ADD CONSTRAINT Kupuje_bilet FOREIGN KEY (ID_odwiedzajacego)
398 REFERENCES Odwiedzajacy (ID_odwiedzajacego)
399 /
400
401
402
403 ALTER TABLE Atrakcje ADD CONSTRAINT Oferuje_atrakcje FOREIGN KEY (ID_oceanarium)
404 REFERENCES Oceanaria (ID_oceanarium)
405 /
406
407
408
409 ALTER TABLE Zbiornik ADD CONSTRAINT Opiekuje_sie FOREIGN KEY (ID_pracownika)
410 REFERENCES Pracownicy (ID_pracownika)
411 /
412
413
414
415 ALTER TABLE Pracownicy ADD CONSTRAINT Obsluguje_sklep FOREIGN KEY (ID_sklepu)
416 REFERENCES Sklepy (ID_sklepu)
417 /
418

```

```

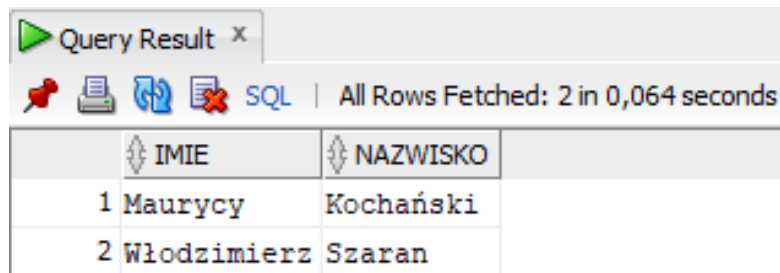
419 |
420 |
421 | ALTER TABLE Zwierzeta ADD CONSTRAINT Ma_zwierze FOREIGN KEY (ID_oceanarium)
422 | REFERENCES Oceanaria (ID_oceanarium)
423 | /
424 |
425 |
426 |
427 | ALTER TABLE Wynagrodzenia ADD CONSTRAINT Pracownik_otrzymuje_wynagrodzenie FOREIGN KEY
428 | (ID_pracownika) REFERENCES Pracownicy (ID_pracownika)
429 | /
430 |
431 |
432 |
433 | ALTER TABLE Odwiedzajacy ADD CONSTRAINT Odwiedzajacy_ma_adres FOREIGN KEY (ID_adresu)
434 | REFERENCES Adresy (ID_adresu)
435 | /
436 |
437 |
438 |
439 | ALTER TABLE Oceanaria ADD CONSTRAINT Oceanarium_ma_adres FOREIGN KEY (ID_adresu)
440 | REFERENCES Adresy (ID_adresu)
441 | /
442 |
443 |
444 |
445 | ALTER TABLE Znajomosci_jezykow_obcych ADD CONSTRAINT Jezyk_jest_znany FOREIGN KEY
446 | (ID_jezyka) REFERENCES Jezyki (ID_jezyka)
447 | /
448 |
449 |
450 |
451 | ALTER TABLE Znajomosci_jezykow_obcych ADD CONSTRAINT Przewodnik_zna_jezyki FOREIGN KEY
452 | (ID_pracownika) REFERENCES Przewodnicy (ID_pracownika)
453 | /
454 |
455 |
456 |
457 | ALTER TABLE Pracownicy ADD CONSTRAINT Pracownik_ma_adres FOREIGN KEY (ID_adresu)
458 | REFERENCES Adresy (ID_adresu)
459 | /

```

#### 5.4. Przykłady zapytań i poleceń SQL odnoszących się do bazy danych

- Informacje o pracownikach będącymi opiekunami płci męskiej.

```
1 SELECT IMIE, NAZWISKO
2 FROM pracownicy
3 WHERE plec='M' AND stanowisko='Opiekun'
```



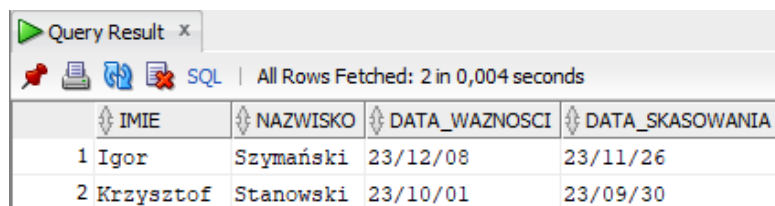
Query Result x

SQL | All Rows Fetched: 2 in 0,064 seconds

	IMIE	NAZWISKO
1	Maurycy	Kochański
2	Włodzimierz	Szaran

- Informacja o gościach oceanariów którzy odwiedzili obiekt na podstawie daty skasowania ich biletu.

```
1 SELECT IMIE, NAZWISKO, DATA_WAZNOSCI, DATA_SKASOWANIA
2 FROM odwiedzajacy o INNER JOIN bilety b ON o.id_odwiedzajacego = b.id_odwiedzajacego
3 WHERE b.data_skasowania IS NOT NULL
```



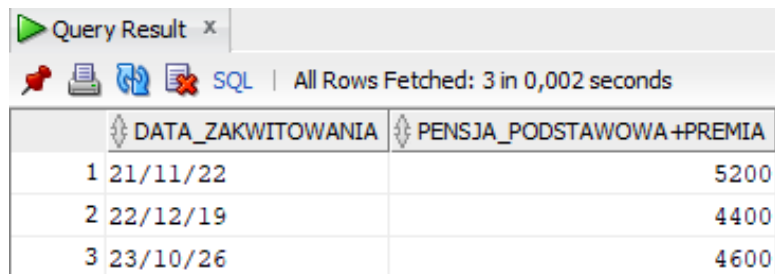
Query Result x

SQL | All Rows Fetched: 2 in 0,004 seconds

	IMIE	NAZWISKO	DATA_WAZNOSCI	DATA_SKASOWANIA
1	Igor	Szymański	23/12/08	23/11/26
2	Krzysztof	Stanowski	23/10/01	23/09/30

- Informacje o dacie i całkowitej ilości wypłaconego wynagrodzenia sortowane od najstarszego.

```
1 SELECT DATA_ZAKWITOWANIA, PENSJA_PODSTAWOWA+PREMIA
2 FROM WYNAGRODZENIA
3 ORDER BY DATA_ZAKWITOWANIA ASC
```



Query Result x

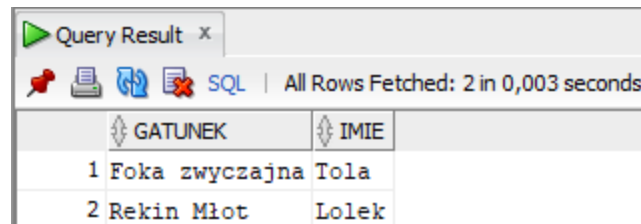
SQL | All Rows Fetched: 3 in 0,002 seconds

	DATA_ZAKWITOWANIA	PENSJA_PODSTAWOWA+PREMIA
1	21/11/22	5200
2	22/12/19	4400
3	23/10/26	4600



[•] Informacje o zwierzętach żyjących w zbiornikach o pojemności powyżej 200 tysięcy litrów.

```
1 SELECT GATUNEK, IMIE
2 FROM ZWIERZETA Z
3 WHERE Z.ID_ZBIORNIKA IN (SELECT ID_ZBIORNIKA FROM ZBIORNIKI
4 WHERE POJEMNOSC > 200000)
```



The screenshot shows a 'Query Result' window with a toolbar containing icons for a pin, print, refresh, and SQL editor. The status bar indicates 'All Rows Fetched: 2 in 0,003 seconds'. The result is displayed in a table with two columns: 'GATUNEK' and 'IMIE'. The first row shows 'Foka zwyczajna' and 'Tola', and the second row shows 'Rekin Młot' and 'Lolek'.

	GATUNEK	IMIE
1	Foka zwyczajna	Tola
2	Rekin Młot	Lolek

## 6. Bibliografia

Do sporządzenia dokumentacji pomocniczo skorzystaliśmy ze slajdów z przedmiotu Bazy Danych i Big Data (BDBT) oraz z udostępnionego nam gotowego projektu bazy danych autorstwa dr hab. inż. Marcina Kowalczyka (Instytut Telekomunikacji na Wydziale Elektroniki i Technik Informacyjnych Politechniki Warszawskiej).