

# Dokumentacja — projekt bazy danych

Czerwiec 2025

## 1. Spis użytych technologii:

1. Jupyter Notebook – użyty do połączenia skryptów generujących dane i wypełniających tabele oraz opisów poszczególnych tabel.
2. Python (wersja 3.12.1) – użyty do napisania skryptów. Wykorzystano następujące biblioteki:
  - a) NumPy
  - b) Pandas
  - c) Collections
  - d) Typing
  - e) Datetime
3. R (wersja 4.4.1) – użyty do przeprowadzenia analizy danych. Wykorzystano następujące pakiety:
  - a) RMariaDB
  - b) Ggplot2
  - c) Knitr
4. Quarto – użyty do połączenia skryptów w języku R oraz komentarzy zawierających treść raportu, a następnie wygenerowania gotowego raportu.
5. LaTeX – użyty do napisania dokumentacji.

## 2. Lista plików i opis ich zawartości:

1. StarWars.ipynb – plik Jupyter Notebook z podstawowymi informacjami o firmie, opisem warunków, z których korzystaliśmy przy wypełnianiu tabel danymi oraz skryptami napisanymi w języku Python, do generowania oraz wypełniania danymi bazy.
2. customer\_names.csv – plik CSV, zawierający listę imion i nazwisk potencjalnych klientów, pobrany ze strony [Kaggle](#).
3. StarWars.json.vuerd – schemat bazy danych ERD w formacie Vuerd.
4. Raport.qmd – plik Quarto z tekstem raportu oraz skryptami w języku R, służącymi do połączenia się z bazą danych oraz uzyskania wyników zapytań, a także do analizy otrzymanych wyników.
5. Raport.html – plik wynikowy, otrzymany po zrenderowaniu pliku Quarto, zawierający analizę danych z bazy.
6. Dokumentacja\_bazy\_danych.pdf - dokumentacja projektu w formacie PDF.

## 3. Kolejność i sposób uruchamiania kolejnych plików, aby uzyskać gotowy projekt:

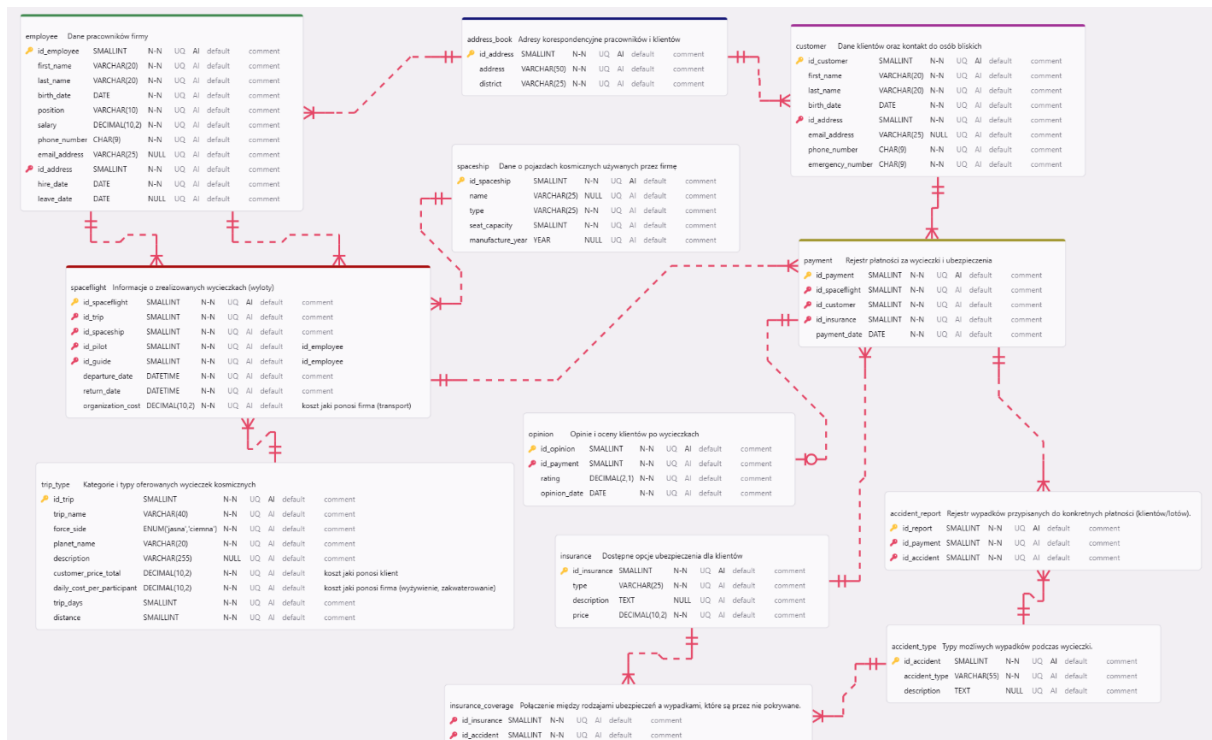
W celu uruchomienia wszystkich plików, niezbędne jest posiadanie Pythona w wersji przynajmniej 3.11 oraz R w wersji 4.4.1. wraz z wymienionymi bibliotekami i pakietami, a także odpowiednie środowiska programistyczne. Niezbędna będzie także możliwość obsługi plików Jupyter Notebook oraz Quarto.

Kolejność uruchamiania plików:

1. W pierwszej kolejności należy otworzyć plik StarWars.ipynb. Uruchomienie wszystkich komórek opcją Run All pozwoli na połączenie się z bazą danych, następnie stworzenie odpowiednich tabel oraz wypełnienie ich danymi według określonych warunków.

- Następnie należy otworzyć plik Raport.qmd, zawierający tekst raportu oraz komórki ze skryptem napisanym w języku R, służącym do połączenia z bazą danych oraz wykonania zdefiniowanych zapytań. Wykonanie opcji „Render” pozwoli na wywołanie każdej komórki kodu oraz uzyskanie wyników, a także stworzenie gotowego raportu w formacie HTML.
- Po utworzeniu pliku Raport.html dostępny jest tekst raportu, wykonanego na podstawie analizy danych z bazy.

#### 4. Schemat projektu bazy danych:



Rysunek 1. Schemat projektu bazy danych

#### 5. Lista zależności funkcyjnych z wyjaśnieniem dla każdej relacji:

##### 5.1. Lista zależności funkcyjnych w tabelach

- employee**:  $id\_employee \rightarrow first\_name, last\_name, birth\_date, position, salary, phone\_number, email\_address, id\_address, hire\_date, leave\_date$   
 $id\_employee$  – klucz główny,  $id\_address$  – klucz obcy.  
 Id pracownika wskazuje jednoznacznie na konkretną osobę, którą opisują imię, nazwisko, data urodzenia i inne dane personalne, a także data zatrudnienia i (ewentualnie) odejścia z firmy, zajmowane stanowisko i otrzymywana wypłata.
- address\_book**:  $id\_address \rightarrow address, district$   
 $id\_address$  – klucz główny.  
 Każdy adres ma przypisane w tabeli unikalne id.
- customer**:  $id\_customer \rightarrow first\_name, last\_name, birth\_date, id\_address, email\_address, phone\_number, emergency\_number$   
 $id\_customer$  – klucz główny,  $id\_address$  – klucz obcy.  
 Podobnie jak w przypadku tabeli **employee**, id klienta wskazuje jednoznacznie na konkretną osobę o określonych danych personalnych.
- spaceship**:  $id\_spaceship \rightarrow name, type, seat\_capacity, manufacture\_year$   
 $id\_spaceship$  – klucz główny.

- Każdy posiadany przez firmę pojazd kosmiczny charakteryzuje się pewną nazwą, typem, pojemnością i rokiem produkcji.
5. **spaceflight**:  $id\_spaceflight \rightarrow id\_trip, id\_spaceship, id\_pilot, id\_guide, departure\_date, return\_date, organisation\_cost$   
 $id\_spaceflight$  – klucz główny,  $id\_trip, id\_spaceship, id\_pilot, id\_guide$  – klucze obce.  
 Każdy wylot związany jest z pewną wycieczką, pojazdem, pilotem, przewodnikiem, datą wyjazdu oraz powrotu i kosztem organizacji.
  6. **payment**:  $id\_payment \rightarrow id\_spaceflight, id\_customer, id\_insurance, payment\_date$   
 $id\_payment$  – klucz główny,  $id\_spaceflight, id\_customer, id\_insurance$  – klucze obce.  
 Każda płatność związana jest z określonym lotem, jest uiszczana przez pewną osobę, która wykupiła ubezpieczenie lub nie, oraz jest dokumentowana określonego dnia.
  7. **trip\_type**:  $id\_trip \rightarrow trip\_name, force\_side, planet\_name, description, customer\_price\_total, daily\_cost\_per\_participant, trip\_days, distance$   
 $id\_trip$  – klucz główny.  
 Każda wycieczka ma inny typ, przypisaną stronę mocy, nazwę planety i opis, wiąże się z określonymi kosztami, trwa pewną ilość dni, a w jej trakcie przebywana jest określona odległość.
  8. **opinion**:  $id\_opinion \rightarrow id\_payment, rating, opinion\_date$   
 $id\_opinion$  – klucz główny,  $id\_payment$  – klucz obcy.  
 Określona opinia została wystawiona po danej wycieczce, wyraża pewną ocenę i została wystawiona konkretnego dnia.
  9. **insurance**:  $id\_insurance \rightarrow type, description, price$   
 $id\_insurance$  – klucz główny.  
 Każde ubezpieczenie ma przypisany typ, opis i cenę.
  10. **insurance\_coverage**:  $id\_accident, id\_insurance$   
 $id\_accident, id\_insurance$  – klucze obce.  
 Różne wypadki są przyporządkowane do różnych typów ubezpieczeń.
  11. **accident\_report**:  $id\_report \rightarrow id\_payment, id\_accident$   
 $id\_report$  – klucz główny,  $id\_payment, id\_accident$  – klucze obce.  
 Każdy zgłoszony wypadek łączy się z osobą, która dokonała danej rezerwacji i uległa określonemu wypadkowi.
  12. **accident\_type**:  $id\_accident \rightarrow accident\_type, description$   
 $id\_accident$  – klucz główny.  
 Id wypadku wskazuje jednoznacznie na typ i opis wypadku.

## 5.2. Lista zależności pomiędzy tabelami

- **employee** i **spaceflight**:  $employee.id\_employee \rightarrow spaceflight.id\_pilot$   
 Relacja 1:N (jeden do wielu, pracownik może latać jako pilot na wiele wycieczek).
- **employee** i **spaceflight**:  $employee.id\_employee \rightarrow spaceflight.id\_guide$   
 Relacja 1:N (jeden do wielu, pracownik może latać jako przewodnik na wiele wycieczek).
- **address\_book** i **employee**:  $address\_book.id\_address \rightarrow employee.id\_address$   
 Relacja 1:N (jeden do wielu, jeden adres może być przypisany do wielu pracowników).
- **address\_book** i **customer**:  $address\_book.id\_address \rightarrow customer.id\_address$   
 Relacja 1:N (jeden do wielu, jeden adres może być przypisany do wielu klientów).
- **spaceship** i **spaceflight**:  $spaceship.id\_spaceship \rightarrow spaceflight.id\_spaceship$   
 Relacja 1:N (jeden do wielu, jeden statek odbył wiele lotów).
- **trip\_type** i **spaceflight**:  $trip\_type.id\_trip \rightarrow spaceflight.id\_trip$   
 Relacja 1:N (jeden do wielu, każdy typ wycieczki mógł odbyć się wielokrotnie).
- **spaceflight** i **payment**:  $spaceflight.id\_spaceflight \rightarrow payment.id\_spaceflight$   
 Relacja 1:N (jeden do wielu, na tą samą wycieczkę poleciało wiele osób).
- **customer** i **payment**:  $customer.id\_customer \rightarrow payment.id\_customer$   
 Relacja 1:N (jeden do wielu, jeden klient mógł wielokrotnie pojechać na wycieczkę).
- **payment** i **opinion**:  $payment.id\_payment \rightarrow opinion.id\_payment$   
 Relacja 0:1 (zero lub jeden, po każdej wycieczce można wystawić co najwyżej jedną opinię).
- **insurance** i **payment**:  $insurance.id\_insurance \rightarrow payment.id\_insurance$   
 Relacja 1:N (jeden do wielu, jeden typ ubezpieczenia mógł zostać wykupiony w wielu rezerwacjach).
- **insurance** i **insurance\_coverage**:

*insurance.id\_insurance* → *insurance\_coverage.id\_insurance*

Relacja 1:N (jeden do wielu, jeden typ ubezpieczenia pokrywa kilka wypadków).

— **accident\_type i insurance\_coverage:**

(*insurance\_coverage.id\_insurance*, *insurance\_coverage.id\_accident*)

Relacja N:M (wiele do wielu, jeden typ wypadku jest pokrywany przez wiele ubezpieczeń, jedno ubezpieczenie może obejmować wiele typów wypadków).

— **accident\_type i accident\_report:**

*accident\_type.id\_accident* → *accident\_report.id\_accident*

Relacja 1:N (jeden do wielu, każdy typ wypadku został zgłoszony wielokrotnie).

— **accident\_type i accident\_report:**

*accident\_type.id\_payment* → *accident\_report.id\_payment*

Relacja 1:N (jeden do wielu, każdy typ wypadku został zgłoszony w przypadku wielu rezerwacji).

## 6. Uzasadnienie, że baza jest w EKNF:

Baza jest w EKNF, ponieważ każda z komórek zawiera pojedynczą wartość i wszystkie rekordy są unikalne (baza spełnia wymogi 1NF), w tabelach nie ma częściowych zależności (baza spełnia wymogi 2NF) i nie zawierają one zależności przechodnich (spełniony wymóg 3 NF). Ponadto wartości w poszczególnych polach tabel zależą tylko od klucza głównego. W przypadku tabeli *insurance\_coverage* klucz traktujemy jako parę – nie istnieje żadna tabela, która odwoływałaby się tylko do prawej lub lewej kolumny. Dlatego baza danych jest w EKNF.

## 7. Opis, co było najtrudniejsze podczas realizacji projektu:

Najtrudniejsza w realizacji projektu była część związana z tworzeniem tabel, zwłaszcza w kwestii stworzenia spójnego systemu opłat za wycieczki (m.in. za zakwaterowanie czy paliwo) oraz stworzenie tabel z informacjami na temat ubezpieczeń tak, aby baza danych spełniała również wymogi EKNF. Część ta wymagała dobrego przemyślenia schematu bazy danych oraz warunków, branych pod uwagę przy wstawianiu wartości.