**Bazy danych - sprawozdanie z projektu**

# Podstawowe informacje

* **Autorzy projektu:** Mateusz Marciniak, Oskar Wojtkowiak
* **Nazwa grupy:** Elektroniczny Biznes
* **Temat:** Magazyn
* **Wybrany zakres (ocena):** Bardzo Dobry
* **Baza danych:**

1. Nazwa bazy danych (serwer venusdb): IE88245
2. Hasło: MM88245

# Spis treści

[1. Podstawowe informacje 1](#_Toc188814115)

[2. Spis treści 2](#_Toc188814116)

[3. Wprowadzenie – opis studium przypadku 3](#_Toc188814117)

[4. Model związków encji 4](#_Toc188814118)

[5. Model relacyjnej bazy danych 4](#_Toc188814119)

[6. Opis utworzonych tabel wraz z poleceniami DDL i DML 5](#_Toc188814120)

[7. Przypadki użycia bazy danych 8](#_Toc188814121)

[8. Pozostałe obiekty bazy danych 12](#_Toc188814122)

# Wprowadzenie – opis studium przypadku

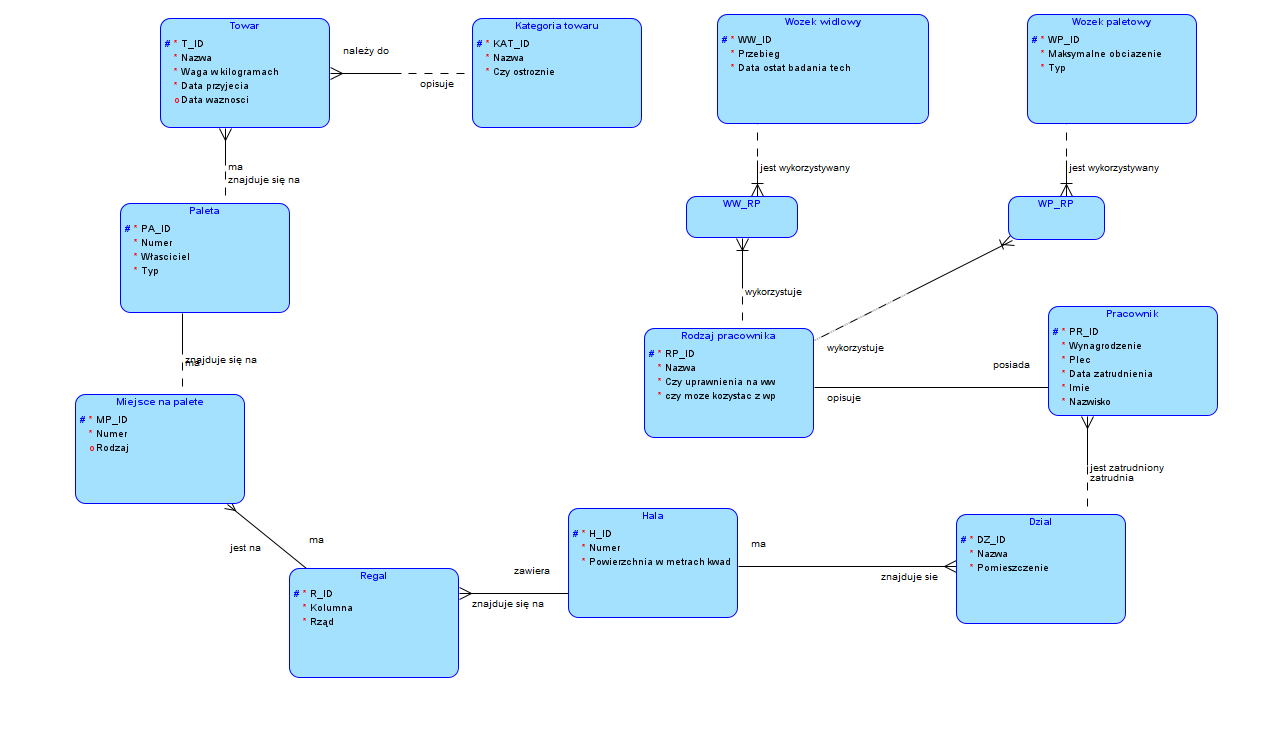
Niniejsze sprawozdanie dotyczy projektu z zakresu relacyjnych baz danych, którego celem jest zaprojektowanie i wdrożenie systemu bazodanowego wspierającego zarządzanie magazynem. Projekt obejmuje zarówno opracowanie graficznego modelu związków encji (ERM), jak i implementację relacyjnej bazy danych w języku SQL z wykorzystaniem poleceń DDL, DML oraz zapytań SELECT. Przygotowany system ma na celu zwiększenie efektywności operacyjnej poprzez poprawę organizacji zapasów oraz automatyzację kluczowych procesów magazynowych.

Projektowany system umożliwi kompleksowe zarządzanie towarami, w tym ich lokalizowanie, przyjmowanie oraz wydawanie. Opracowana baza danych będzie wspierać kontrolowanie stanów magazynowych, ułatwiać identyfikację potencjalnych braków oraz wspomagać podejmowanie decyzji związanych z optymalizacją logistyki. W projekcie uwzględniono zarówno wymagania funkcjonalne, takie jak obsługa zapasów i generowanie raportów, jak i niefunkcjonalne, obejmujące bezpieczeństwo, skalowalność oraz wydajność systemu.

W ramach prac nad projektem przygotowano szczegółowy opis studium przypadku, obejmujący analizę wymagań biznesowych oraz technicznych. Model związków encji został przekształcony w model relacyjny, a wszystkie tabele bazy danych wypełniono realistycznymi danymi, odzwierciedlającymi rzeczywiste procesy zachodzące w magazynie. W sprawozdaniu przedstawiono także osiem przypadków użycia, które ilustrują potencjalne zastosowania bazy danych w kontekście operacyjnym, wraz z odpowiednimi zapytaniami SQL wykorzystującymi zaawansowane techniki, takie jak złączenia tabel, funkcje grupowe oraz podzapytania.

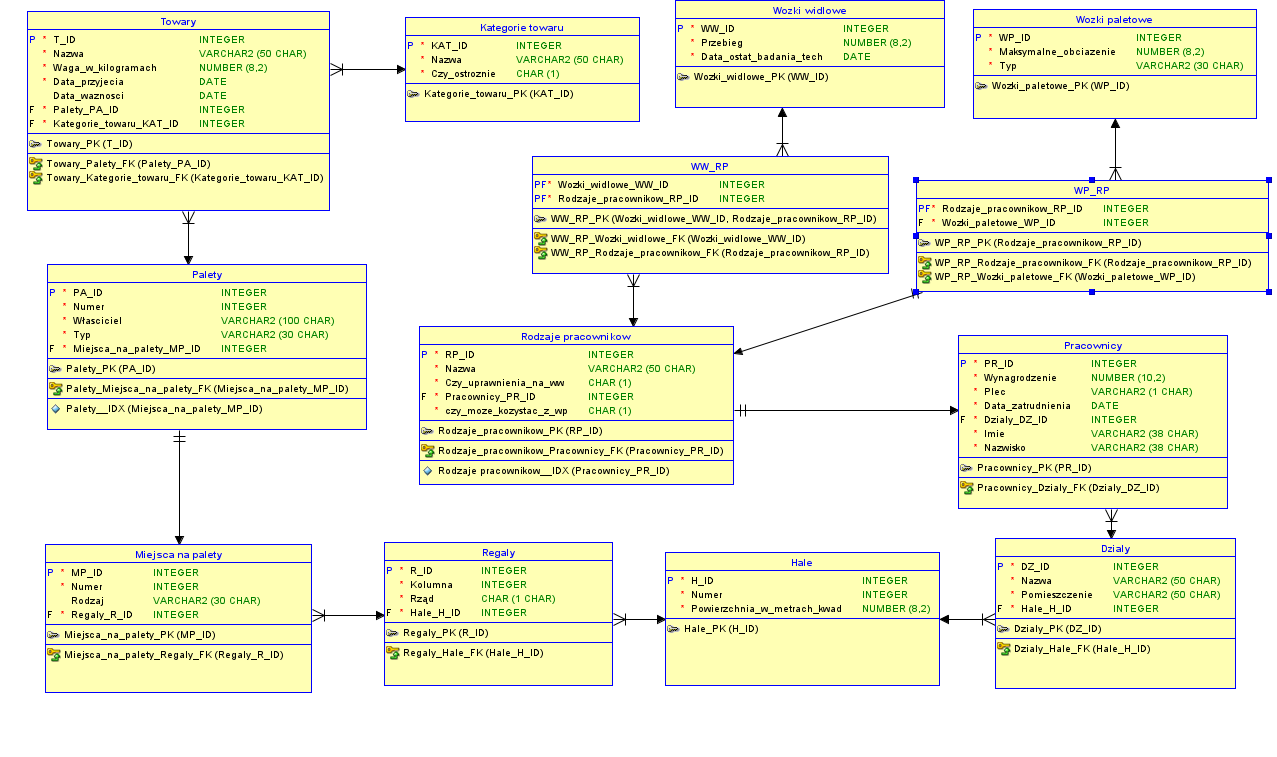
Całość prac została zrealizowana zgodnie z wymogami projektowymi, a wyniki zostały zapisane w formie kopii zapasowej bazy danych, co umożliwia ich późniejsze wykorzystanie. Projekt stanowi przykład efektywnego wykorzystania technologii bazodanowych w celu optymalizacji procesów zarządzania magazynem, a przygotowana dokumentacja pełni rolę szczegółowego opisu technicznego wdrożonego rozwiązania.

# Model związków encji



[**(źródło)**](../projekt_bd_zal_1_magazyn_e/model_encje_fin.png)

# Model relacyjnej bazy danych



[**(źródło)**](../projekt_bd_zal_1_magazyn_e/model_relacyjny_fin.png)

[**(Spis treści)**](#_Spis_treści)

# Opis utworzonych tabel wraz z poleceniami DDL i DML

**dzialy**

* **Opis**: Reprezentuje działy zlokalizowane w halach magazynowych.
* **Kolumny**:
  + dz\_id: Unikalny identyfikator działu (klucz główny).
  + nazwa: Nazwa działu.
  + pomieszczenie: Numer pomieszczenia, w którym znajduje się dział.
  + hale\_h\_id: Identyfikator hali (klucz obcy do tabeli hale).
* **Klucz główny**: dzialy\_pk na kolumnie dz\_id.
* **Klucz obcy**: dzialy\_hale\_fk odwołujący się do tabeli hale.

**Hale**

* **Opis**: Przechowuje informacje o halach magazynowych.
* **Kolumny**:
  + h\_id: Unikalny identyfikator hali (klucz główny).
  + numer: Numer hali.
  + powierzchnia\_w\_metrach\_kwad: Powierzchnia hali w metrach kwadratowych.
* **Klucz główny**: hale\_pk na kolumnie h\_id.

**Kategoria Towaru**

* **Opis**: Określa kategorie towarów przechowywanych w magazynie.
* **Kolumny**:
  + kat\_id: Unikalny identyfikator kategorii (klucz główny).
  + nazwa: Nazwa kategorii.
  + czy\_ostroznie: Flaga oznaczająca, czy towary z tej kategorii wymagają ostrożnego traktowania.
* **Klucz główny**: kategorie\_towaru\_pk na kolumnie kat\_id.

**Miejsca na palety**

* **Opis**: Określa lokalizacje na regałach, gdzie przechowywane są palety.
* **Kolumny**:
  + mp\_id: Unikalny identyfikator miejsca na palety (klucz główny).
  + numer: Numer miejsca.
  + rodzaj: Typ miejsca.
  + regaly\_r\_id: Identyfikator regału (klucz obcy do tabeli regaly).
* **Klucz główny**: miejsca\_na\_palety\_pk na kolumnie mp\_id.
* **Klucz obcy**: miejsca\_na\_palety\_regaly\_fk odwołujący się do tabeli regaly.

**Palety**

* **Opis**: Reprezentuje palety przechowywane w magazynie.
* **Kolumny**:
  + pa\_id: Unikalny identyfikator palety (klucz główny).
  + numer: Numer palety.
  + własciciel: Nazwa właściciela palety.
  + typ: Typ palety.
  + miejsca\_na\_palety\_mp\_id: Identyfikator miejsca na palety (klucz obcy).
* **Klucz główny**: palety\_pk na kolumnie pa\_id.
* **Klucz obcy**: palety\_miejsca\_na\_palety\_fk odwołujący się do tabeli Miejsca na palety.
* **Indeks unikalny**: Na kolumnie miejsca\_na\_palety\_mp\_id.

**pracownicy**

* **Opis**: Przechowuje informacje o pracownikach.
* **Kolumny**:
  + pr\_id: Unikalny identyfikator pracownika (klucz główny).
  + wynagrodzenie: Wynagrodzenie pracownika.
  + plec: Płeć pracownika (kod literowy, np. "M" lub "K").
  + data\_zatrudnienia: Data zatrudnienia pracownika.
  + dzialy\_dz\_id: Identyfikator działu (klucz obcy).
  + Rodzaje pracownikow\_RP\_ID: Identyfikator rodzaju pracownika (klucz obcy).
* **Klucz główny**: pracownicy\_pk na kolumnie pr\_id.
* **Klucze obce**: pracownicy\_dzialy\_fk, pracownicy\_rodzaje\_fk.
* **Indeks unikalny**: Na kolumnie Rodzaje pracownikow\_RP\_ID.

**regaly**

* **Opis**: Reprezentuje regały w halach magazynowych.
* **Kolumny**:
  + r\_id: Unikalny identyfikator regału (klucz główny).
  + kolumna: Numer kolumny regału.
  + rząd: Numer rzędu regału.
  + hale\_h\_id: Identyfikator hali (klucz obcy).
* **Klucz główny**: regaly\_pk na kolumnie r\_id.
* **Klucz obcy**: regaly\_hale\_fk odwołujący się do tabeli hale.

**Rodzaje pracownikow**

* **Opis**: Przechowuje informacje o rodzajach pracowników.
* **Kolumny**:
  + rp\_id: Unikalny identyfikator rodzaju pracownika (klucz główny).
  + nazwa: Nazwa rodzaju pracownika.
  + czy\_uprawnienia\_na\_ww: Flaga oznaczająca, czy pracownik ma uprawnienia do obsługi wózka widłowego.
  + pracownicy\_pr\_id: Identyfikator pracownika (klucz obcy).
  + czy\_moze\_kozystac\_z\_wp: Flaga oznaczająca, czy pracownik może korzystać z wózka paletowego.
* **Klucz główny**: rodzaje\_pracownikow\_pk na kolumnie rp\_id.
* **Klucz obcy**: rodzaje\_pracownikow\_fk.

**towary**

* **Opis**: Przechowuje informacje o towarach w magazynie.
* **Kolumny**:
  + t\_id: Unikalny identyfikator towaru (klucz główny).
  + nazwa: Nazwa towaru.
  + waga\_w\_kilogramach: Waga towaru.
  + data\_przyjecia: Data przyjęcia towaru.
  + data\_waznosci: Data ważności towaru.
  + palety\_pa\_id: Identyfikator palety (klucz obcy).
  + kategorie\_towaru\_kat\_id: Identyfikator kategorii towaru (klucz obcy).
* **Klucz główny**: towary\_pk na kolumnie t\_id.
* **Klucze obce**: towary\_palety\_fk, towary\_kategorie\_towaru\_fk.

**Wozki paletowe i Wozki widlowe**

* **Opis**: Przechowują informacje o wózkach paletowych i widłowych.
* **Kolumny**:
  + Wózki paletowe:
    - wp\_id: Unikalny identyfikator wózka paletowego (klucz główny).
    - maksymalne\_obciazenie: Maksymalne obciążenie wózka.
    - typ: Typ wózka.
  + Wózki widłowe:
    - ww\_id: Unikalny identyfikator wózka widłowego (klucz główny).
    - przebieg: Przebieg wózka w kilometrach.
    - data\_ostat\_badania\_tech: Data ostatniego badania technicznego.

**Polecenia DDL i DML**

[**(Odwołanie tutaj)**](../projekt_bd_zal_1_magazyn_e/sql_komendy_projekt.txt)

[**(Spis Treści)**](#_Spis_treści)

# Przypadki użycia bazy danych

1. Use Case: Łączenie tabel / Funkcja grupowa / Grupowanie danych / Sortowanie

-- Opis: To zapytanie liczy liczbę pracowników oraz oblicza średnie wynagrodzenie w każdym dziale.

-- Zawiera:

-- 1. Łączenie tabel: "pracownicy" i "dzialy" (JOIN).

-- 2. Funkcje grupowe: COUNT (liczba pracowników), AVG (średnie wynagrodzenie).

-- 3. Grupowanie danych: po nazwie działu.

-- 4. Sortowanie danych: wyniki posortowane alfabetycznie po nazwie działu.

SELECT d.nazwa AS dzial,

       COUNT(p.PR\_ID) AS liczba\_pracownikow,

       AVG(p.Wynagrodzenie) AS srednie\_wynagrodzenie

FROM pracownicy p

JOIN dzialy d ON p.Dzialy\_DZ\_ID = d.dz\_id

GROUP BY d.nazwa

ORDER BY d.nazwa;

-- 2. Use Case: Łączenie tabel / Funkcja grupowa / Grupowanie danych

-- Opis: Zapytanie liczy liczbę wózków widłowych przypisanych do każdego pracownika.

-- Zawiera:

-- 1. Łączenie tabel: "pracownicy", "Rodzaje pracownikow", "ww\_rp" oraz "Wozki widlowe".

-- 2. Funkcja grupowa: COUNT (liczba wózków widłowych).

-- 3. Grupowanie danych: po imieniu i nazwisku pracownika.

SELECT p.Imie, p.Nazwisko, COUNT(ww.ww\_id) AS liczba\_wozkow

FROM pracownicy p

JOIN "Rodzaje pracownikow" rp ON p.PR\_ID = rp.pracownicy\_pr\_id

JOIN ww\_rp wrp ON rp.rp\_id = wrp.rodzaje\_pracownikow\_rp\_id

JOIN "Wozki widlowe" ww ON wrp.wozki\_widlowe\_ww\_id = ww.ww\_id

GROUP BY p.Imie, p.Nazwisko;

-- 3. Use case:  Funkcja jednowierszowa numeryczna i funkcja obsługująca NULL

-- Opis: Zapytanie wyświetla imiona, nazwiska i wynagrodzenia pracowników.

-- Jeśli wynagrodzenie jest NULL, zostaje zastąpione przez 'Brak'.

-- Zawiera:

-- 1. Funkcja obsługująca NULL: CASE (zastąpienie NULL tekstem 'Brak').

-- 2. Funkcja numeryczna: ROUND (zaokrąglenie wynagrodzenia).

-- 3. Funkcja konwersji typów: TO\_CHAR (przekształcenie wynagrodzenia na tekst).

SELECT p.Imie, p.Nazwisko,

       CASE

           WHEN p.Wynagrodzenie IS NULL THEN 'Brak' -- Zastąpienie NULL tekstem 'Brak'

           ELSE TO\_CHAR(ROUND(p.Wynagrodzenie, 2)) -- Wartość wynagrodzenia jako tekst, zaokrąglona do 2 miejsc po przecinku

       END AS wynagrodzenie, -- Funkcja obsługująca NULL i zaokrąglająca

       d.nazwa AS dzial

FROM pracownicy p

JOIN dzialy d ON p.Dzialy\_DZ\_ID = d.dz\_id -- JOIN na tabeli pracownicy i dzialy

WHERE p.Wynagrodzenie IS NOT NULL; -- Warunek, aby uwzględnić tylko pracowników z wynagrodzeniem

-- 4. Use Case: Łączenie tabel, Funkcje numeryczne, Grupowanie danych

-- Opis: Zapytanie sumuje powierzchnię regałów w działach, obliczając łączną powierzchnię regałów.

-- Zawiera:

-- 1. Łączenie tabel: "hale", "regaly" oraz "dzialy".

-- 2. Funkcje numeryczne: SUM (łączna powierzchnia regałów).

-- 3. Grupowanie danych: po nazwie działu.

SELECT d.nazwa AS dzial, SUM(h.powierzchnia\_w\_metrach\_kwad) AS laczna\_powierzchnia

FROM hale h

JOIN regaly r ON h.h\_id = r.hale\_h\_id

JOIN dzialy d ON r.hale\_h\_id = d.hale\_h\_id

GROUP BY d.nazwa;

-- 5. Use Case: Funkcja do obsługi dat, Łączenie tabel, Funkcja konwersji typów danych, Grupowanie danych

-- Opis: To zapytanie liczy liczbę towarów przyjętych w określonym okresie oraz ich kategorie.

-- Zawiera:

-- 1. Łączenie tabel: "towary" oraz "Kategorie towaru".

-- 2. Funkcja do obsługi dat: TO\_DATE (konwersja tekstu na datę).

-- 3. Funkcja konwersji typów danych: TO\_DATE (konwersja tekstu na datę).

-- 4. Grupowanie danych: po nazwie towaru i kategorii towaru.

SELECT t.nazwa AS towar, kt.nazwa AS kategoria, COUNT(t.t\_id) AS liczba\_towarow

FROM towary t

JOIN "Kategorie towaru" kt ON t.kategorie\_towaru\_kat\_id = kt.kat\_id

WHERE t.data\_przyjecia BETWEEN TO\_DATE('2022-01-01', 'YYYY-MM-DD') AND TO\_DATE('2024-01-31', 'YYYY-MM-DD')

GROUP BY t.nazwa, kt.nazwa;

-- 6. Use Case: Łączenie tabel, Funkcja numeryczna, Grupowanie danych

-- Opis: To zapytanie sumuje powierzchnię regałów w działach, podobnie jak w przypadku 4.

-- Zawiera:

-- 1. Łączenie tabel: "hale", "regaly", "dzialy".

-- 2. Funkcje numeryczne: SUM (łączna powierzchnia regałów).

-- 3. Grupowanie danych: po nazwie działu.

SELECT d.nazwa AS dzial, SUM(h.powierzchnia\_w\_metrach\_kwad) AS laczna\_powierzchnia

FROM hale h

JOIN regaly r ON h.h\_id = r.hale\_h\_id

JOIN dzialy d ON r.hale\_h\_id = d.hale\_h\_id

GROUP BY d.nazwa;

-- 7. Use Case: Łączenie tabel, Sortowanie

-- Opis: Zapytanie wyświetla pracowników w danym dziale (pomieszczenie '101') posortowanych według daty zatrudnienia.

-- Zawiera:

-- 1. Łączenie tabel: "pracownicy" oraz "dzialy".

-- 2. Sortowanie danych: według daty zatrudnienia.

SELECT p.Imie, p.Nazwisko, p.Data\_zatrudnienia

FROM pracownicy p

JOIN dzialy d ON p.Dzialy\_DZ\_ID = d.dz\_id

WHERE d.pomieszczenie = '101'

ORDER BY p.Data\_zatrudnienia;

-- 8 Use Case: Podzapytanie nieskorelowane

-- Opis: Zapytanie wyświetla pracowników, których wynagrodzenie jest wyższe niż średnia wynagrodzeń we wszystkich firmach.

-- Zawiera: Podzapytanie nieskorelowane, JOIN, SELECT, WHERE.

SELECT p.Imie, p.Nazwisko, p.Wynagrodzenie

FROM pracownicy p

WHERE p.Wynagrodzenie > (

    SELECT AVG(Wynagrodzenie) -- Nieskorelowane podzapytanie

    FROM pracownicy

);

-- 9 Use Case: Podzapytanie skorelowane i funkcja znakowa

-- Opis: Zapytanie wyświetla pracowników, którzy zarabiają więcej niż średnia wynagrodzenia w ich dziale.

-- Funkcja znakowa: Dodanie "Panie/Pani" przed imieniem pracownika w zależności od płci.

SELECT

    CASE

        WHEN p.Plec = 'M' THEN 'Pan ' || p.Imie    -- Funkcja znakowa dla mężczyzn

        WHEN p.Plec = 'K' THEN 'Pani ' || p.Imie   -- Funkcja znakowa dla kobiet

        ELSE p.Imie

    END AS imie\_zmodyfikowane,  -- Modyfikacja imienia

    p.Nazwisko,                  -- Nazwisko

    p.Wynagrodzenie              -- Wynagrodzenie

FROM

    pracownicy p

WHERE

    p.Wynagrodzenie >= (

        SELECT AVG(p2.Wynagrodzenie)  -- Podzapytanie skorelowane

        FROM pracownicy p2

        WHERE p.Dzialy\_DZ\_ID = p2.Dzialy\_DZ\_ID  -- Skorelowane po dziale

    );

-- 10 use case:  Łączenie 5 tabel

-- Opis: Zapytanie zlicza liczbę pracowników w każdym dziale oraz średnie wynagrodzenie w danym dziale

-- Zawiera: Łączenie tabel, funkcję COUNT i AVG.

SELECT

    d.nazwa AS dzial,                  -- Nazwa działu

    COUNT(p.PR\_ID) AS liczba\_pracownikow, -- Liczba pracowników w dziale

    AVG(p.Wynagrodzenie) AS srednie\_wynagrodzenie -- Średnie wynagrodzenie w dziale

FROM

    dzialy d

JOIN

    pracownicy p ON p.Dzialy\_DZ\_ID = d.dz\_id   -- Łączenie z tabelą pracownicy

JOIN

    "Rodzaje pracownikow" rp ON rp.pracownicy\_pr\_id = p.PR\_ID -- Łączenie z tabelą rodzaje pracowników

JOIN

    wp\_rp wprp ON wprp.rodzaje\_pracownikow\_rp\_id = rp.rp\_id -- Łączenie z tabelą wp\_rp

JOIN

    "Wozki paletowe" wp ON wp.wp\_id = wprp.wozki\_paletowe\_wp\_id -- Łączenie z tabelą wozki\_paletowe

GROUP BY

    d.nazwa;  -- Grupowanie po nazwie działu

[**(Odwołanie tutaj)**](../venusdb.sql)

[**(Spis treści)**](#_Spis_treści)

# Pozostałe obiekty bazy danych

opis pozostałych obiektów bazy danych takich jak perspektywy, sekwencery, synonimy.

W projekcie zastosowano tylko sekwencery.

**seq\_towary**

* Tworzy sekwencer o nazwie seq\_towary, który zaczyna liczyć od 1 i inkrementuje o 1.
* Jest używany w tabeli towary do przypisania wartości domyślnej do kolumny t\_id:

CREATE TABLE towary (

t\_id INTEGER DEFAULT seq\_towary.NEXTVAL NOT NULL,

nazwa VARCHAR2(50 CHAR) NOT NULL,

waga\_w\_kilogramach NUMBER(8, 2) NOT NULL,

data\_przyjecia DATE NOT NULL,

data\_waznosci DATE,

palety\_pa\_id INTEGER NOT NULL,

kategorie\_towaru\_kat\_id INTEGER NOT NULL

);

**seq\_pracownicy**

* Ten kod tworzy sekwencer o nazwie seq\_pracownicy, który będzie zaczynał się od wartości 1 i zwiększał się o 1 za każdym razem, gdy zostanie użyty do generowania kolejnej wartości (np. dla kolumny typu INTEGER, która będzie korzystać z tego sekwencera).
* Dzięki temu, każda nowa wartość generowana przez ten sekwencer będzie unikalna, a sekwencer będzie automatycznie inkrementował licznik po każdej operacji generowania nowej wartości.

CREATE SEQUENCE seq\_pracownicy START WITH 1 INCREMENT BY 1;

[**(Odwołanie tutaj)**](../projekt_bd_zal_1_magazyn_e/sql_komendy_projekt.txt)

[**(Spis Treści)**](#_Spis_treści)