

# Mapowanie obiektowo relacyjne



# Agenda

1. Wstęp
2. relacyjne bazy danych
3. relacje w językach obiektowych
4. mapowanie relacji na obiekty
5. ćwiczenia

# Relacyjne bazy danych

- baza danych w formie tabel i połączeń między nimi
- tabele składają się z wierszy i kolumn
- każde pole w tabeli ma określony typ
- każdy wiersz posiada identyfikator unikalny względem danej tabeli
- obsługa transakcyjności
- SQL - język zapytań
- RDBMS - system zarządzania bazą danych

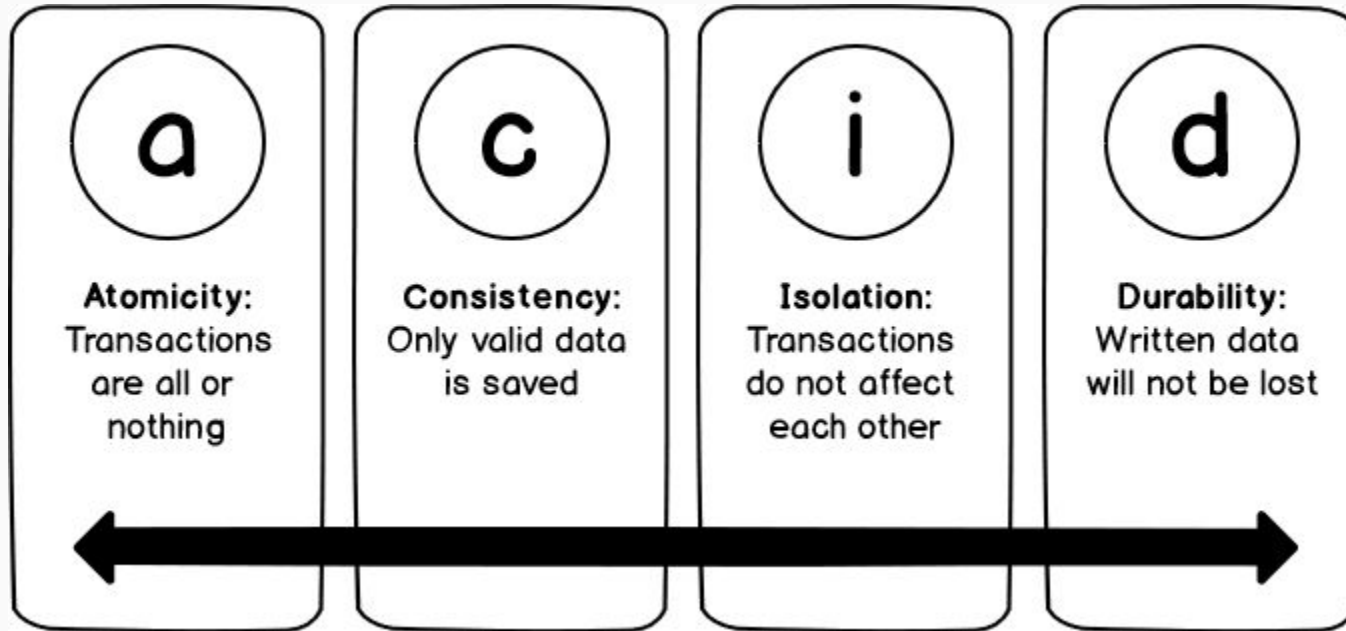
# Relacyjne bazy danych - przykłady

- PostgreSQL
- SQL Server
- Oracle SQL
- RDS - Amazon Relational Database Service

# Transakcje

- łączenie wielu pojedynczych operacji w jedną operację logiczną - “single unit of work”
- zapewniane bezpośrednio przez bazę danych
- ACID

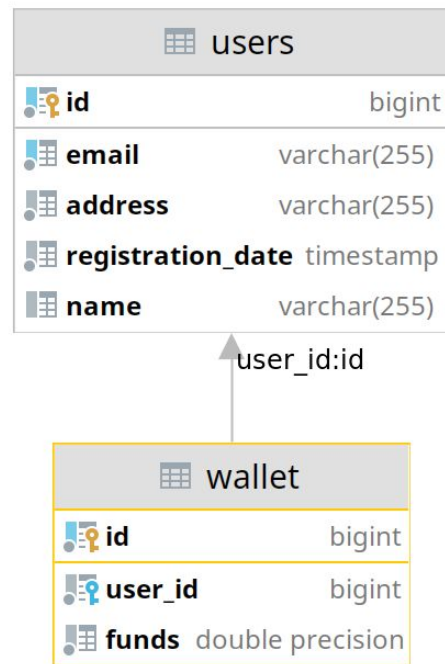
# Transakcije - ACID



# Zależności między obiektami 1/3

## Relacja jeden do jednego

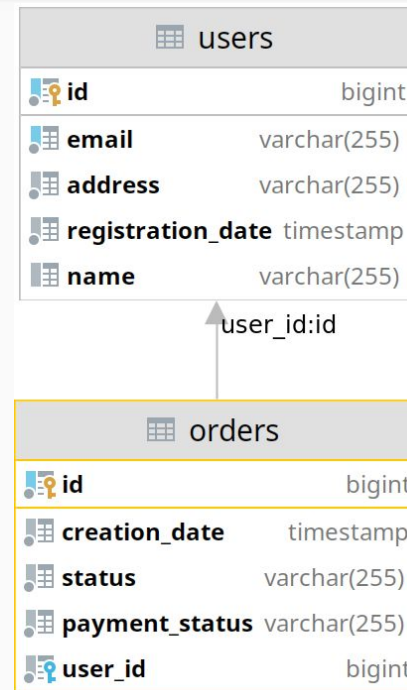
- @OneToOne
- w językach obiektowych: referencja do obiektu
- w bazie danych: klucz obcy



# Zależności między obiektami 2/3

## Relacja jeden do wielu

- @OneToMany i @ManyToOne
- w językach obiektowych: kolekcja obiektów
- w bazie danych: klucz obcy

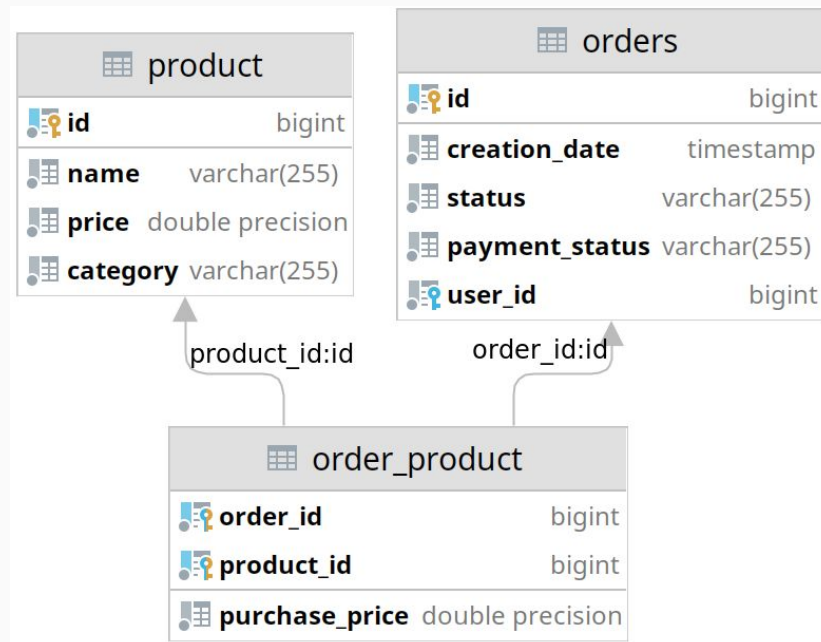




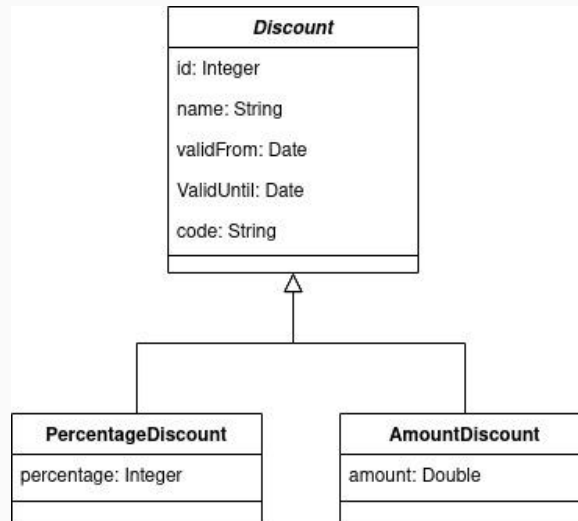
# Zależności między obiektami 3/3

## Relacja wiele do wielu

- @ManyToMany lub złożenie @OneToMany i @ManyToOne
- w językach obiektowych: kolekcja obiektów
- w bazie danych: tabela pośrednia










# Relacja dziedziczenia



# Mapowanie relacji dziedziczenia 1/3

## Typ mapowania **Single Table**

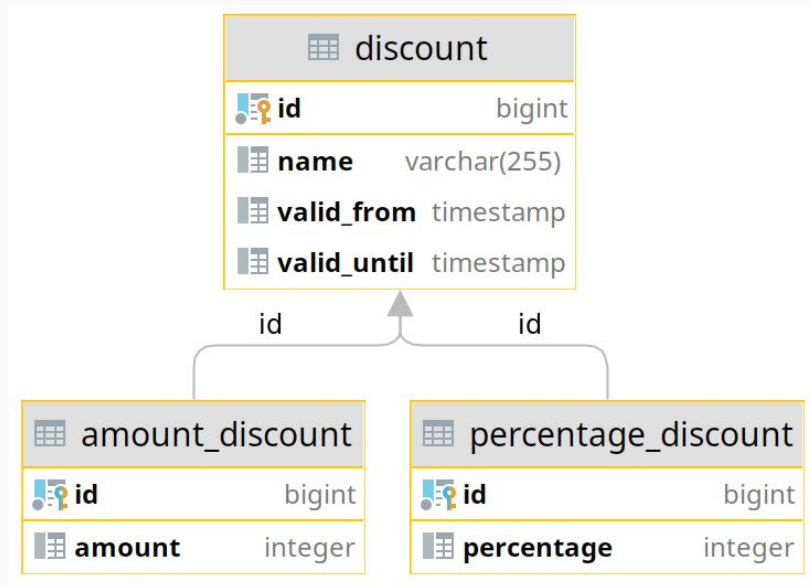
- tworzona jest tylko jedna tabela
- pola klas dziedziczących mogą mieć wartości null (i na pewno będą)
- konieczne jest dodatkowe pole z konkretnym typem

discount	
 <b>id</b>	bigint
 <b>dtype</b>	varchar(31)
 <b>name</b>	varchar(255)
 <b>valid_from</b>	timestamp
 <b>valid_until</b>	timestamp
 <b>amount</b>	integer
 <b>percentage</b>	integer

# Mapowanie relacji dziedziczenia 2/3

## Typ mapowania **Class Table**

- oddzielna tabela dla klasy bazowej
- oddzielna tabela dla każdej klasy implementującej, ale zawierająca pola tylko tej klasy + id



# Mapowanie relacji dziedziczenia 3/3

## Typ mapowania **Concrete Class Table**

- każda klasa ma swoją tabelę ze wszystkimi polami
- pola są zduplikowane

percentage_discount	
id	bigint
name	varchar(255)
valid_from	timestamp
valid_until	timestamp
percentage	integer

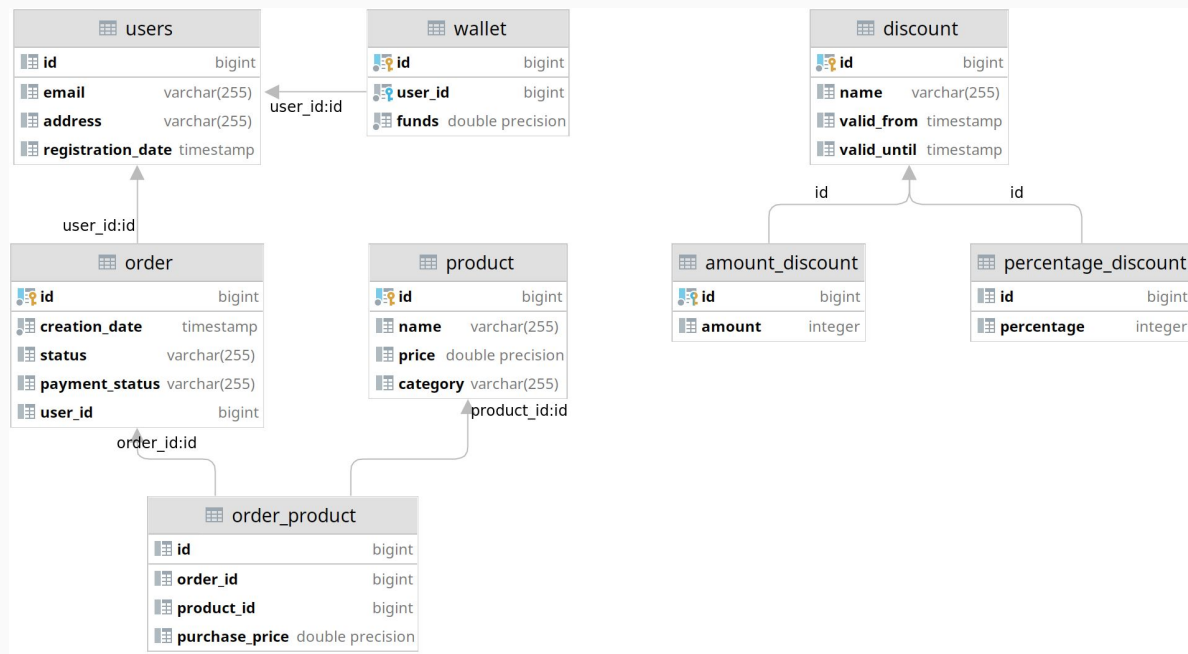
amount_discount	
id	bigint
name	varchar(255)
valid_from	timestamp
valid_until	timestamp
amount	integer

discount	
id	bigint
name	varchar(255)
valid_from	timestamp
valid_until	timestamp

# ORM w Javie - JPA

- JPA (Java Persistence Api) - ogólny interfejs do ORM
- najbardziej znana implementacja - Hibernate
- część standardu JavaEE
- wykorzystywane również w Springu
- tabele są reprezentowane przez obiekty Entity

# Zadanie 1 - implementacja schematu



# Zadanie 2 - implementacja REST API

Wymagania:

- API do pobierania produktów wyszukiwanych po nazwie
- API do pobierania niezakończonych zamówień dla użytkownika o danym id (sumowanie ceny produktów)
- autentykacja użytkownika
- API CRUD do zniżek



# Zadanie 3 - implementacja zakupu produktu

Wymagania:

- Implementacja API
- możliwość wyboru produktów
- pobranie środków z portfela
- zasymulowanie błędu i przetestowanie cofania transakcji

# Źródła

- <https://www.techopedia.com/definition/16455/transaction-databases>
- <https://devopedia.org/acid-transactions>