Multimedialne i Obiektowe Bazy Danych

dr Artur Niewiadomski

Wykład 1

Wprowadzenie

Ewolucja modeli danych

Literatura

- A. Barczak, M. Barczak, Projektowanie i implementacja bazy dokumentów, Wydawnictwo Naukowe UPH, 2020.
- P. Saladge, M. Fowler, NoSQL. Kompendium wiedzy. Helion 2014
- A. Wiśniewski, Aspekty obiektowości w obiektowo-relacyjnym systemie Oracle 9i/10g. Wyd. AP, Siedlce 2007
- A. Barczak, A. Wiśniewski, Podstawy multimedialnych systemów baz danych. Vizja Press&IT,Warszawa 2009
- P. Saladge, M. Fowler, NoSQL. Kompendium wiedzy. Helion 2014
- J. Dickey, Nowoczesne aplikacje internetowe : MongoDB, Express, AngularJS, Node.js, Helion 2016

Literatura uzupełniająca

- D.Sullivan. NoSQL. Przyjazny przewodnik. Helion 2016
- M. McLaughlin, Oracle database 12c. Programowanie w języku PL/SQL. Helion 2015

Tematyka wykładów

- Wprowadzenie do problematyki modeli danych oraz systemów multimedialnych i obiektowych baz danych

Tematyka wykładów

- Wprowadzenie do baz NoSQL.
- Przegląd rodziny baz NoSQL.
- Bazy klucz-wartość.
- Bazy dokumentów.
- Bazy kolumnowe.
- Grafowe bazy danych.
- Wybrane bazy NoSQL w praktyce.
- MongoDB. Neo4J.

Tematyka wykładów

- Charakterystyka standardu obiektowych baz danych ODMG
- Elementy obiektowości w architekturze obiektowo-relacyjnej (na przykładzie Oracle)
- Składowanie i charakterystyka danych multimedialnych
- Eksploracja i obsługa danych multimedialnych
- Standardy SQL/MM, MPEG-7

Modele danych

Model danych

- Złożenie, wypadkowa kilku cech, takich jak:
  - Terminologia (metajęzyk)
  - Języki opisu i przetwarzania danych
  - Ogólne założenia dotyczące architektury systemu baz danych
  - Teorie i ograniczenia dotyczące struktur danych i dostępu do danych

Podział modeli danych

- Podstawowe typy (generacje) modeli danych:
  - Proste
- Pliki
  - Klasyczne
- Hierarchiczne, sieciowe, relacyjne
  - Semantyczne
- Semistrukturalne, obiektowe
  - Obiektowo – relacyjne

Proste modele danych

- Struktura plików płaskich (flat files)
- Rekordy przechowywane w plikach
- Pliki zorganizowane liniowo, sekwencyjnie
- Brak relacji między plikami
- Ograniczone operacje (odczyt, zapis)
- Dziś spotykane dość rzadko, w starych aplikacjach

Proste modele danych

- Wraz z rozwojem urządzeń do przechowywania danych wprowadzono różne optymalizacje, np.
- Podstawowa metoda bezpośredniego dostępu BDAM (Basic Direct Access Method)
- Algorytm mieszający (hashing algorithm) generuje adres rekordu na dysku
- Bardzo szybki dostęp
- Duże zużycie przestrzeni dyskowej

Proste modele danych

- Metoda indeksowanego dostępu sekwencyjnego ISAM (Indexed Sequential Access Method)
- Dodatkowe pliki z indeksami zawierające klucze rekordów i ich adresy dyskowe
- Efektywne wykorzystanie przestrzeni dyskowej

Klasyczne (tradycyjne) modele danych

- Hierarchiczne
- Sieciowe
- Relacyjne

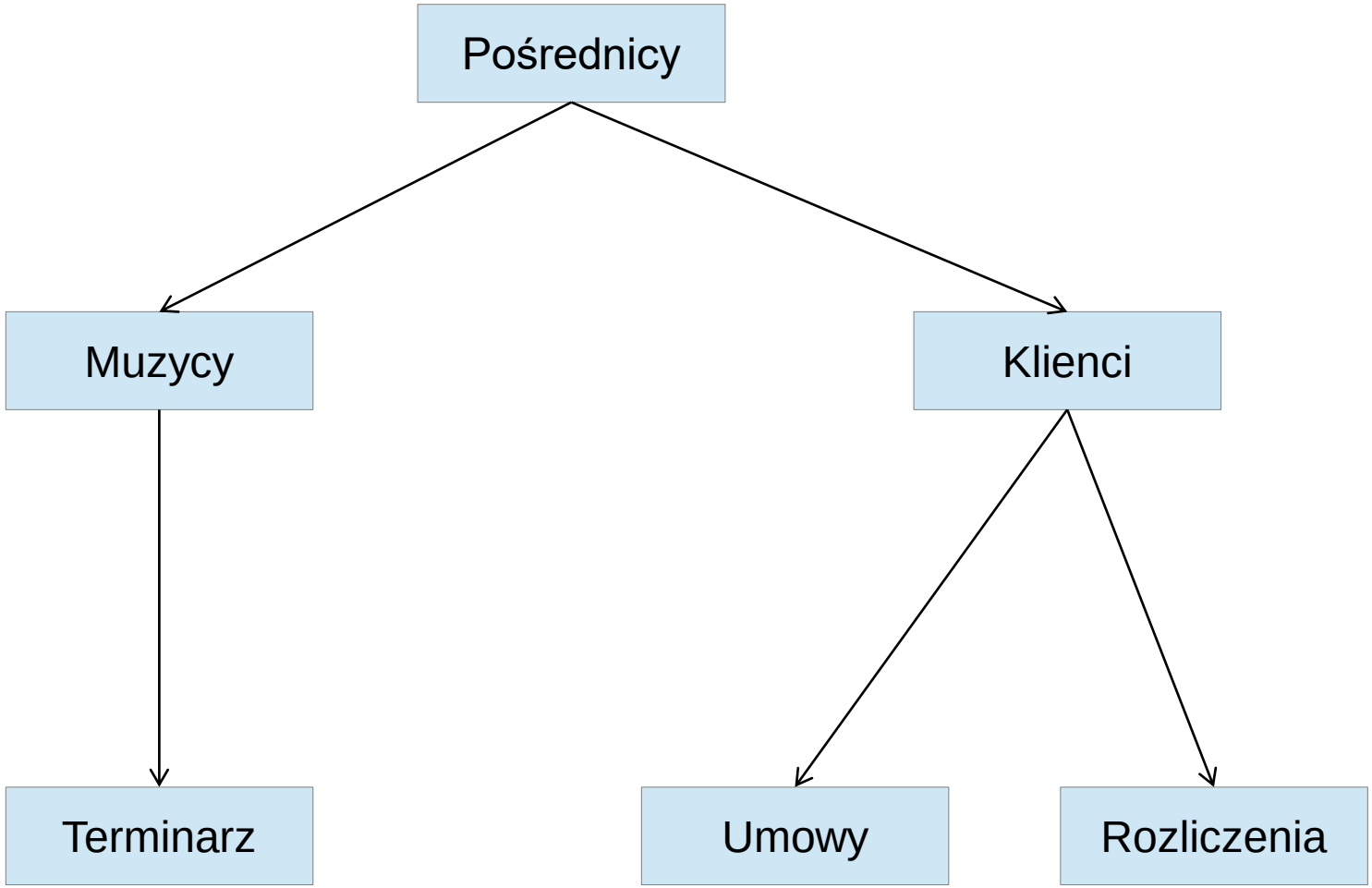
Hierarchiczny model danych

- Rozszerzenie modelu prostego do struktury drzewiastej
- Wprowadzenie typu rekordu
- Relacje typu nadrzędny-podrzędny
- Integralność danych zapewniona przez strukturę
- Dostęp do niższych warstw struktury tylko przez warstwy nadrzędne

Hierarchiczny model danych

- Główne zalety:
  - łatwość zastosowania,
  - duża szybkość działania,
- Wady:
  - ścisłe reguły dotyczące relacji,
  - wstawianie i kasowanie danych może okazać się bardzo skomplikowane,
  - dostęp do niższych warstw jest możliwy tylko poprzez warstwy nadrzędne,
  - trudności w modelowaniu relacji typu wiele-do-

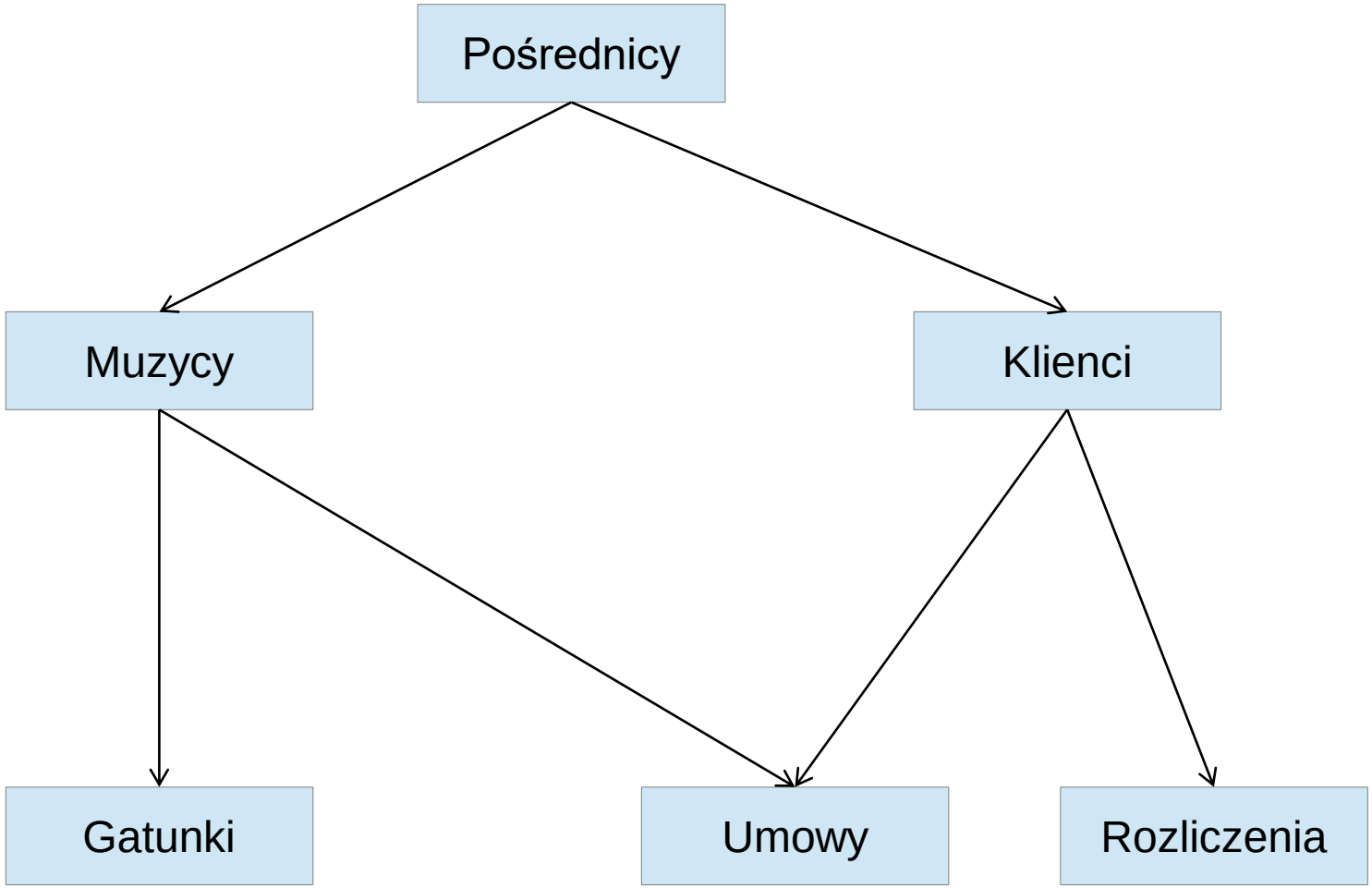
Przykład modelu hierarchicznego



Sieciowy model danych

- Zaproponowany przez organizację CODASYL
- Rozszerzenie modelu hierarchicznego
- Struktura danych tworzy graf, sieć
- Węzły i gałęzie mogą być współdzielone
- Referencyjne lub wskaźnikowe powiązania
- Kolekcje
- Języki definiowania i manipulowania danymi

Przykład modelu sieciowego



Sieciowy model danych

- Zalety:
  - Przeszukiwanie od dowolnego miejsca,
  - Przeglądanie danych w górę i w dół
  - Możliwość tworzenia bardziej złożonych zapytań niż w HMBD
- Wady:
  - Wymagana ścisła znajomość struktury bazy
  - Trudności przy zmianach strukturalnych

Relacyjny model danych

- Zaproponowany przez E.F. Codd'a
- Oparty na teorii mnogości i rachunku predykatów pierwszego rzędu
- Zasadniczy cel: poprawa poziomu niezależności programu od danych
- Obecnie najczęściej stosowany model

Relacyjny model danych

- Dane przechowywane w tabelach (relacjach)
- Tabele to dwuwymiarowe tablice składające się z wierszy i kolumn
- Wiersze to inaczej krotki lub rekordy, a kolumny to atrybuty tabeli
- Tabele niezależne od siebie, inaczej niż w modelu hierarchicznym i sieciowym

Relacyjny model danych

- Podstawowe zasady związane z tabelami:
  - Jednoznaczność nazwy tabeli
  - Jednoznaczność nazwy kolumny w obrębie tabeli
  - Identyczny typ danych w całej kolumnie
  - Atomowość wartości zawartej na przecięciu wiersza i kolumny

Relacyjny model danych

- Tabela posiada jedno lub kilka pól tworzących klucz główny, który jednoznacznie identyfikuje wiersz
- Powiązania między tabelami realizowane za pomocą kluczy obcych
- Wartością klucza obcego jest wartość pewnego klucza głównego

Relacyjny model danych

- Zalety
  - Prostota.
  - Niezależność danych – możliwe modyfikacje struktury danych bez wpływu na istniejące programy.
  - Deklaratywny język dostępu do danych. Użytkownik określa jedynie warunki, system natomiast zajmuje się pobraniem danych spełniających żądanie. Nawigacja ukryta przed użytkownikiem.

Relacyjny model danych

- Wady
  - Brak złożonych obiektów,
  - Brak możliwości przechowywania informacji proceduralnych
  - Niezgodność impedancji

Model relacyjny – niezgodność impedancji

- SQL nie jest pełnym językiem programowania, więc wymaga zanurzenia w uniwersalny język programowania
- Potrzebne są konstrukcje, które umożliwiają takie zanurzenie
- Konsekwencją połączenia dwóch różnych języków jest niekorzystny efekt określany jako niezgodność impedancji.

Niezgodność impedancji, czyli niezgodność w zakresie

- Składni
  - programista musi w jednym tekście programu używać dwóch stylów językowych i przestrzegać reguł dwóch różnych gramatyk
- Systemu typów
  - język zapytań operuje na typach zdefiniowanych w schemacie bazy danych, m.in. relacjach, natomiast język programowania posiada zwykle odmienny system typów, w którym nie występuje typ relacja.

Niezgodność impedancji, czyli niezgodność w zakresie

- Semantyki i paradygmatów języków
  - język zapytań bazuje na stylu deklaratywnym, a języki programowania zazwyczaj bazują na stylu imperatywnym.
- Poziomu abstrakcji
  - Język zapytań uwalnia programistę od wielu szczegółów organizacji i implementacji danych (np. organizacji zbiorów, obecności lub nieobecności indeksów itd.), podczas gdy w języku programowania te szczegóły muszą być

Niezgodność impedancji, czyli niezgodność w zakresie

- Faz i mechanizmów wiązania.
  - Języki zapytań są oparte na późnym wiązaniu (są interpretowane), podczas gdy języki programowania często zakładają wczesne wiązanie (podczas kompilacji i konsolidacji).
- Przestrzeni nazw i reguł zakresu.
  - Język zapytań i język programowania posiadają własne przestrzenie nazw, które mogą zawierać identyczne nazwy o różnych znaczeniach. Odwzorowania pomiędzy przestrzeniami nazw

Niezgodność impedancji, czyli niezgodność w zakresie

- Traktowania wartości zerowych.
  - Bazy danych i języki zapytań posiadają wyspecjalizowane środki dla przechowywania i przetwarzania wartości zerowych. Środki te nie występują w językach programowania.
- Schematów iteracyjnych.
  - W języku zapytań iteracje są wtopione w semantykę operatorów, takich jak selekcja, projekcja i złączenie. W języku programowania iteracje muszą być organizowane explicite za

Niezgodność impedancji, czyli niezgodność w zakresie

- Traktowania cechy trwałości danych.
  - Języki zapytań przetwarzają wyłącznie trwałe dane (znajdujące się na dysku), podczas gdy języki programowania przetwarzają wyłącznie dane nietrwałe znajdujące się w pamięci operacyjnej. Połączenie obu języków wymaga od programisty użycia specjalnych środków językowych do parametryzacji zapytań przez zmienne języka programowania oraz środków językowych i architektonicznych służących do transmisji danych z dysku do pamięci

Niezgodność impedancji, czyli niezgodność w zakresie

- Środków programowania generycznego.
  - Środki te w języku zapytań są oparte na refleksji (patrz np. dynamiczny SQL). Użycie podobnego środka w języku programowania jest zazwyczaj niemożliwe z powodu wczesnego wiązania. Stosowane są inne środki, takie jak funkcje wyższego rzędu, casting, przejście na niższy poziom językowy, polimorfizm lub szablony.

Semantyczne modele danych

- Próbują dostarczyć sposobów reprezentacji ZNACZENIA informacji.
- Model semantyczny składa się z sieci konceptów i relacji pomiędzy tymi konceptami.
- Koncepty są ideami, obiektami lub tematami które interesują użytkownika.
- Koncepty i relacje połączone razem tworzą ontologię - semantyczny model opisujący wiedzę.
- Jednym z prostszych modeli semantycznych jest model obiektowy

Obiektowy model danych

- Paradygmat obiektowy zaaplikowany do świata baz danych
- Koncepcja OBD oparta o zasadnicze elementy technologii obiektowej, takich jak:
  - Pojecie klasy i obiektu
  - Abstrakcja
  - Hermetyzacja
  - Dziedziczenie
  - Polimorfizm

Obiekt

- Abstrakcyjny byt reprezentujący lub opisujący pewną rzecz lub pojęcie w świecie rzeczywistym
- Jest odróżnialny od innych obiektów i ma dobrze określone granice
- Struktura danych przechowywana w pamięci komputera (atrybuty + metody)
- Obiektem można manipulować jako całością

Tożsamość obiektu

- Jednoznaczny, niepowtarzalny identyfikator obiektu (OID)
- Najczęściej generowany przez system
- Obiekty rozróżnialne nawet gdy mają identyczne składniki (atrybuty, metody itp.),
inaczej niż w modelu relacyjnym

Klasa

- Klasa – miejsce przechowywania cech (zestawu atrybutów, nazw metod, ograniczeń dostępu) grupy podobnych obiektów
- W modelu obiektowym obiekty jako instancje klas definiują zawartość bazy,
- Klasy definiują schemat bazy

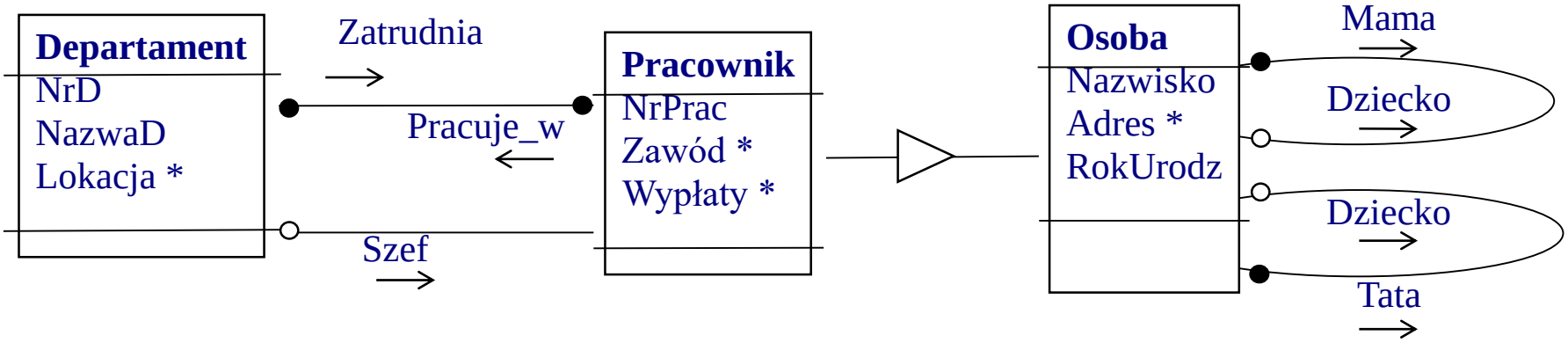
Mechanizmy abstrakcji

- Uogólnienie
  - Deklaracja grupy klas jako podklas innych klas
  - Powiązane z dziedziczeniem
- Agregacja
  - Związek pomiędzy klasami modelujący stosunek całości do części

Obiektowe bazy danych

- Próba eliminacji ograniczeń modelu relacyjnego w zakresie odwzorowań obiektów
- Obsługa danych mających złożoną, zagnieżdżoną strukturę, tworzących hierarchię, dynamicznie zmieniających rozmiar
- Bezpośrednia współpraca z obiektowymi językami programowania

Przykład modelu obiektowego i relacyjnego



Departament(NrD, NazwaD)
Lokacja(NrLokacji, NazwaLok, NrD)
Szef(NrD, NrPrac)
Pracownik(NrPrac, NrOsoby)
PracDept(NrPrac, NrD)
Zawód(IdZawodu, NazwaZawodu, NrPrac)
Wypłata (IdWypłaty,Wysokość, NrPrac)
Osoba(NrOsoby, Nazwisko, RokUrodz)
Adres(NrAdresu, Miejsce, NrOsoby)
Mama(NrOsoby, NrOsoby)
Tata(NrOsoby, NrOsoby)

- Czytelna pojęciowa struktura zamieniła się na 11 schematów relacji
- Pojawiły się nowe atrybuty - klucze
- Semantyka wyrażona poprzez liczności została częściowo zgubiona
- Semantyka dziedziczenia została zgubiona

Manifest OBD

- Cechy obowiązkowe
  - złożone obiekty
  - tożsamość obiektów
  - enkapsulacja
  - dziedziczenie
  - typy lub klasy
  - rozszerzalność
  - zapytania ad hoc
  - kompletność obliczeniowa
  - trwałość
  - współbieżność
  - odtwarzanie
- Cechy opcjonalne
  - wielokrotne dziedziczenie
  - kontrola typów
  - rozproszenie
  - zarządzanie wersjami
- Cechy otwarte
  - paradygmat programowania
  - metody reprezentacji obiektów
  - system typów
  - jednolitość (zgodność)

Systemy obiektowo-relacyjne

- Ostrożna ewolucja systemów relacyjnych
- Adaptacja systemów relacyjnych do obsługi funkcji typowych dla obiektowych baz danych
- Wprowadzenie wielu cech obiektowości, m.in. klas, metod, dziedziczenia
- Dane nadal przechowywane w tabelach, ale ich zawartość jest bogatsza
- Wady: złożoność systemu, niska dojrzałość technologiczna, mała akceptowalność i obszerność standardu SQL3

Porównanie relacyjnych i obiektowych baz danych

zalety

Model relacyjny	Model obiektowy
Dobre mechanizmy kontroli	**Wzbogacone możliwości modelowania**
Niezależność od języka programowania	**Rozszerzalność**
Możliwość zarządzania wielką ilością danych	**Obsługa zmian schematu**
Prostota struktur danych (w pierwotnej wersji modelu)	**Dostosowanie do zaawansowanych zastosowań BD**

Porównanie relacyjnych i obiektowych baz danych

wady

Model relacyjny	Model obiektowy
Niewielkie możliwości reprezentacji rzeczywistych obiektów	Brak uniwersalnego modelu danych
Przeciążenie semantyczne pojęcia relacja	Brak formalnych standardów
Jednorodna struktura (wartości atomowe na przecięciu wiersza i kolumny)	Optymalizacja zapytań w konflikcie z hermetyzacją
Ograniczony zasób operacji	Brak realizacji perspektyw
Trudności w realizowaniu zapytań rekurencyjnych	Brak mechanizmów bezpieczeństwa
Niewielkie możliwości wyrażania więzów integralności i więzów ogólnych	Złożoność

Modelowanie

Własność	Obiektowo-relacyjny model danych	Obiektowy model danych
Tożsamość obiektu (OID)	Realizowana przez typ REF	Dostępna
Hermetyzacja	Realizowana przez typy użytkownika	Dostępna, ale naruszana w przypadku zapytań
Dziedziczenie	Dostępne	Dostępne
Polimorfizm	Dostępne (wywołanie funkcji użytkownika za pomocą funkcji ogólnych)	Realizacja jak w środowisku obiektowych języków programowania
Obiekty złożone	Realizowane przez typy użytkownika	Dostępne
Związki	Realizowane przez więzy integralności referencyjnej	Dostępne