Wprowadzenie do kursu:
Zaawansowane systemy baz danych

Zbyszko KRÓLIKOWSKI

Politechnika Poznańska, Instytut Informatyki

ul. Piotrowo 2, 60-965 Poznań

e-mail: Zbyszko.Krolikowski@cs.put.poznan.pl

1. Wstęp

Organizacja nowoczesnego społeczeństwa opiera się na szerokim zastosowaniu systemów informatycznych. Stanowią one podstawę systemów bankowych, systemów rezerwacji lotniczej, hotelowej i kolejowej, systemów administracyjnych, gospodarki materiałowej i magazynowej, systemów ewidencji ludności, systemów wspomagania projektowania, itp.

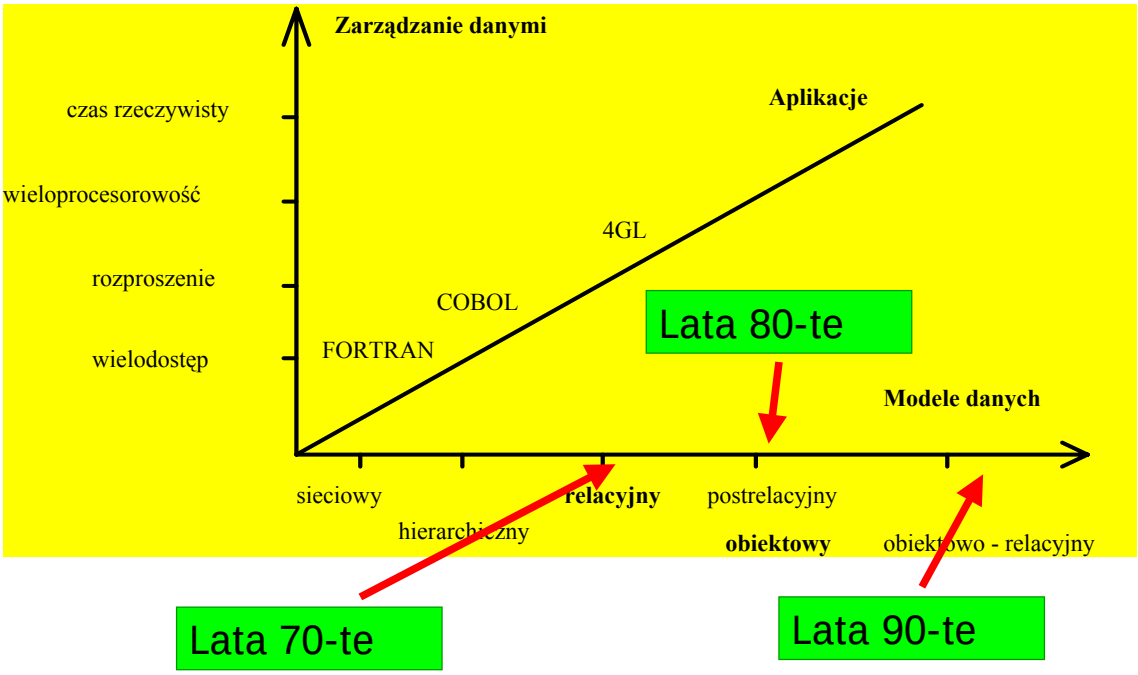
Powszechnie stosowanym narzędziem do tworzenia różnego rodzaju systemów informatycznych są **systemy baz danych** (*SBD*) składające się z **bazy danych** (*BD*) i **systemu zarządzania bazą danych** (*SZBD*).

Najogólniej rzecz biorąc **baza danych** jest komputerową reprezentacją wybranego fragmentu świata rzeczywistego. Przykładami takiego fragmentu są stany kont i dane personalne klientów banku, lub projekt nowego samochodu. Dane w bazie danych reprezentują stan poszczególnych obiektów świata rzeczywistego. Baza danych oprócz właściwych danych zawiera również ich interpretację odzwierciedlającą semantykę obiektów świata rzeczywistego. Do modelowania semantyki świata rzeczywistego w bazie danych są wykorzystywane pewne abstrakcyjne pojęcia składające się na **model danych** bazy danych.

System zarządzania bazą danych jest implementacją konkretnego modelu danych, która powinna charakteryzować się następującymi własnościami funkcjonalnymi.:

- gwarantowanie trwałości i spójności danych w bazie danych,
- zapewnienie współbieżnego dostępu do nich,
- autoryzacja dostępu do bazy danych,
- efektywny dostęp do danych w środowisku scentralizowanym i rozproszonym.

Historię rozwoju systemów baz danych przedstawiono na rys. 1. Początki baz danych sięgają lat 60-tych ubiegłego stulecia, kiedy to pojawiły się pierwsze implementacje baz danych w bankowości. Systemy te były budowane w oparciu o hierarchiczny lub sieciowy model danych. Modele te były bardzo złożone. Dane były ściśle powiązane z programami, które na nich „operowały". Wykorzystywano, tak „egzotyczne" z dzisiejszego punktu widzenia języki programowania jak Cobol czy Fortran. Wszystko to powodowało, że bazy danych były dziedziną zastrzeżoną wyłącznie dla odpowiednio przygotowanych specjalistów.



Rys. 1. Historia rozwoju systemów baz danych.

Przełom nastąpił w roku 1970, kiedy to Codd opublikował swoją pierwszą pracę poświęconą założeniom relacyjnych baz danych. Zaproponowany nowy, relacyjny model danych był znacznie prostszy w swoich założeniach w porównaniu z modelem sieciowym czy hierarchicznym, łatwy w interpretacji i implementacji. Przede wszystkim te właśnie cechy relacyjnego modelu danych, spowodowały gwałtowny rozwój tej dyscypliny informatyki. Kilka lat później pojawiły się pierwsze komercyjne SZBD firm IBM i Oracle. W tym samym okresie zaproponowano język SQL dostępu do relacyjnych baz danych. Od tego momentu obserwujemy gwałtowny wzrost implementacji systemów baz danych w różnych dziedzinach zastosowań.

Biorąc pod uwagę historię rozwoju zastosowań baz danych, aplikacje konwencjonalnych relacyjnych *SBD*, które powstały w latach siedemdziesiątych, można scharakteryzować następująco. W latach siedemdziesiątych i osiemdziesiątych podstawową dziedziną zastosowań były aplikacje administracyjno-finansowe. Przykładem aplikacji tego typu są systemy informatyczne dla banków i towarzystw ubezpieczeniowych, systemy rezerwacji miejsc oraz systemy wspomagające gospodarkę finansową i magazynową przedsiębiorstw o dowolnym profilu działalności. Zastosowania te wiąże charakterystyka występujących w nich danych, które cechuje prosta struktura, standardowe typy danych oraz, w ogólności, brak własności dynamicznych. Zastosowania te łączy również model interakcji z bazą danych, który polega na realizacji krótkich, w większości niezależnych, zapytań.

Ekspansja informatyki we współczesnym świecie powoduje rozszerzanie się dziedzin zastosowań *SBD*. W latach osiemdziesiątych, a szczególnie dziewięćdziesiątych, pojawiły się nowe zastosowania o zupełnie odmiennej charakterystyce. Różnią się one nieraz istotnie semantyką odpowiadającego im fragmentu świata rzeczywistego. Na przykład systemy wspomagania projektowania różnią się od systemów bankowych dużo większą złożonością strukturalną danych, większą złożonością własności dynamicznych i innym sposobem interakcji między użytkownikami bazy danych. W ogólności cechą tych nowych zastosowań jest złożona struktura danych, liczne typy związków między danymi, niestandardowe typy danych oraz złożone własności dynamiczne. Interakcja użytkownika z systemem ma znacznie bardziej złożony charakter niż w systemach konwencjonalnych. To już nie krótkie, niezależne zapytania, lecz wiele skomplikowanych, długich – a tym samym czasochłonnych, wzajemnie powiązanych zapytań. Oprócz systemów wspomagania projektowania, przykładami takich zastosowań są

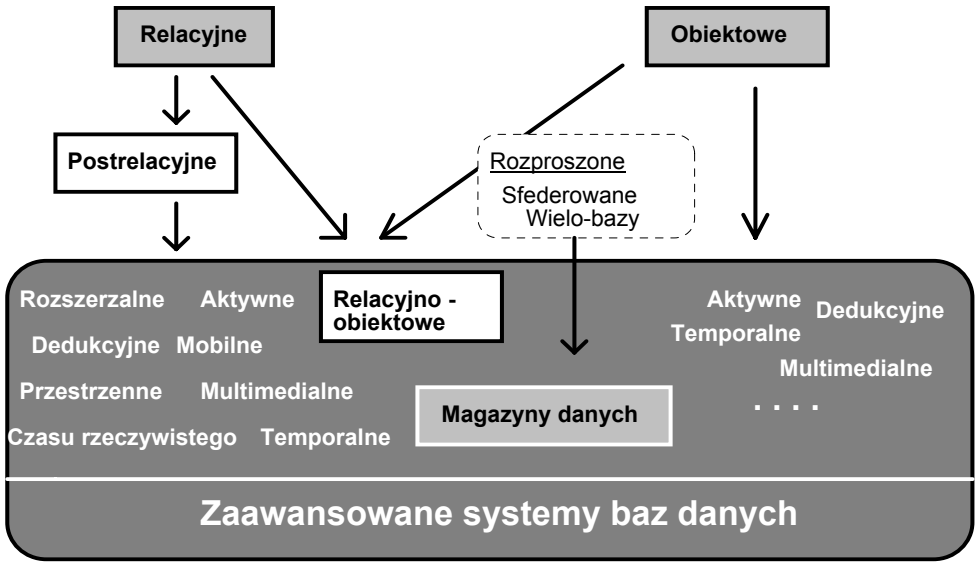
magazyny danych, systemy wspomagania decyzji i produkcji, systemy wspomagania inżynierii oprogramowania, systemy z bazami wiedzy, systemy medyczne i kartograficzne, w których są pamiętane obrazy, zdjęcia i mapy.

Pojawiające się nowe zastosowania baz danych pociągnęły za sobą konieczność poszukiwania nowych modeli danych lub rozszerzeń modeli istniejących, i co za tym idzie nowych rozwiązań w ramach *SZBD*. W ostatnich kilkunastu latach, wychodząc naprzeciw wymogom tych nowych aplikacji, są prowadzone intensywne badania nad wprowadzeniem do *SBD* różnego rodzaju rozszerzeń. Badania te zaowocowały powstaniem tzw. **zaawansowanych systemów baz danych** *(ZSBD)* (ang. *Advanced DBS*).

Niniejszy tekst stanowi wstęp do cyklu wykładów, w ramach których zostaną szczegółowo przedstawione systemy baz danych zaliczane do grupy zaawansowanych. W ramach kursu przedstawimy Państwu wybrane grupy systemów zaawansowanych baz danych. Natomiast w ramach niniejszego wprowadzającego tekstu zostaną omówione generacje systemów baz danych i wymagania stawiane wobec zaawansowanych systemów baz danych oraz zostanie przedstawiona krótka charakterystyka niektórych systemów omawianych w trakcie kursu i systemów, które ze względu na ograniczenia czasowe, w trakcie tego cyklu wykładów omawiane nie będą (post-relacyjne, mobilne). Studiując ten tekst możecie Państwo odnieść wrażenie, że jest on trochę chaotyczny. To prawda. Jednakże moim celem nie było usystematyzowanie tej problematyki, a jedynie prezentacja wybranych, najciekawszych problemów badawczych i implementacyjnych, aby Państwa zachęcić do podjęcia dogłębnych studiów z tej dziedziny.

2. Generacje systemów baz danych

Zarówno w teorii jak i w praktyce, tj. na rynku komercyjnym, zaproponowano wiele systemów baz danych. Z pewnym uproszczeniem, jeszcze kilkanaście lat temu można było wyróżnić ich dwie podstawowe generacje: **relacyjną i obiektową** (rys. 2). Można powiedzieć, że te dwie grupy systemów, tj. relacyjne i obiektowe stanowią obecnie „klasykę”, jeśli chodzi o systemy baz danych.



Rys. 2. Generacje systemów baz danych

Systemy relacyjnych baz danych, jako „klasyka” doczekały się bardzo bogatego piśmiennictwa i w tym miejscu nie ma potrzeby ich bliższej prezentacji – są szeroko omawiane w ramach pierwszego stopnia studiów.

Próby zastosowania w latach osiemdziesiątych systemów baz danych do komputerowego wspomagania projektowania (ang. *Computer Aided Design – CAD*), komputerowego wspomagania wytwarzania (ang. *Computer Aided Manufacturing – CAM*), systemów wspomagania inżynierii oprogramowania (ang. *Computer Aided Software Engineering – CASE*), geograficznych systemów informacyjnych (ang. *Geographical Information Systems – GIS*) oraz systemów komputerowego wspomagania prac zespołowych (ang. *Computer Supported Cooperative Work – CSCW*), zaowocowały **systemami obiektowych baz danych**. Są one oparte na obiektowo-zorientowanym modelu danych (ang. *object-oriented data model*), w którym kluczowym pojęciem jest pojęcie *obiektu*, który integruje strukturalne i operacyjne własności modelu danych. Obiektowe bazy danych stwarzają możliwość modelowania złożonych typów danych, hierarchii typów i własności dynamicznych.

Po pierwszym okresie rozwoju systemów baz danych, w trakcie którego zasadniczo uporano się z problemami zarządzania dużymi zbiorami danych, efektywnej realizacji żądań użytkowników, niezawodności oraz ochrony, obecnie poszukuje się lepszych sposobów modelowania świata rzeczywistego oraz zwiększania "inteligencji" systemów zarządzania bazami danych. W ramach tego cyklu wykładów zajmiemy się przede wszystkim rozszerzeniami wprowadzanymi do systemów baz danych, które wychodzą naprzeciw wymogom współczesnych nie-standardowych aplikacji i mają na celu zwiększanie "inteligencji" *SBD*.

Wprowadzenie wymienionych niżej rozszerzeń pozwala mówić o zwiększonej inteligencji systemu i jego zakwalifikowaniu do generacji systemów zaawansowanych:

- zapewnienie obsługi obiektów o złożonej strukturze;
- obsługa niesformatowanej informacji, np. tekstów, obrazów, map, video;
- dodatkowe funkcje odpowiednie dla różnych dziedzin zastosowań;
- przechowywanie wiedzy o modelowanym świecie, wyrażonej nie tylko za pomocą faktów (danych), lecz także w postaci reguł,
- umiejętność wnioskowania opartego na zapamiętanych regułach i faktach.

Przedstawimy obecnie krótką charakterystykę wybranych grup zaawansowanych systemów baz danych, w których wymienione powyżej rozszerzenia wprowadzono.

3. Systemy post-relacyjnych baz danych

Systemy post-relacyjnych baz danych nie będą szczegółowo omawiane w ramach naszego kursu. Warto jednak zapoznać się z ich krótką charakterystyką, ponieważ rozwiązania zaproponowane swego czasu w ramach badań nad tą grupą systemów, można obecnie znaleźć w wielu współczesnych, komercyjnych SZBD. Innymi słowy, od systemów post-relacyjnych wszystko się zaczęło.

Próba rozszerzenia relacyjnego modelu danych o elementy ułatwiające modelowanie z natury coraz bardziej skomplikowanej rzeczywistości nowych dziedzin zastosowań zaowocowała powstaniem **systemów post-relacyjnych baz danych**. Były one próbą odparcia bardzo mocnej konkurencji ze strony obiektowych baz danych. Systemy post-relacyjne najczęściej nie były tworzone od podstaw, lecz przez wzbogacenie określonych systemów relacyjnych. Klasycznym przykładem może być tutaj system *Postgres* stanowiący rozszerzenie znanego systemu relacyjnego *Ingres*. W uproszczeniu można powiedzieć, że model danych systemów post-relacyjnych jest „nadzbiorem” koncepcji stosowanych w klasycznym modelu relacyjnym.

Systemy post-relacyjne oferowały kilka dodatkowych konstrukcji, powiększających w znaczący sposób siłę wyrazu ich modelu danych. Są to: abstrakcyjne typy atrybutów i złożone struktury danych, zagnieżdżone relacje, atrybuty proceduralne (wirtualne), dziedziczenie (ang. *inheritance*) zapytania historyczne i rozszerzalne funkcje. Omówimy obecnie te nowe, dodatkowe elementy oferowane swego czasu przez systemy post-relacyjne.

W tego typu systemach (na przykład w systemie *Postgres*) nie używano pojęcia relacji lecz znacznie szerszego pojęcia **klasy**. Klasa jest zbiorem **wystąpień** (inaczej, instancji) obiektów, posiadającym określoną nazwę. Formalnie przez **obiekt** rozumie się dwójkę: **o = (oid, v)**, gdzie **oid** jest unikalnym identyfikatorem obiektu, a **v** jest wartością obiektu. Wartość obiektu, w zależności od jego struktury, może być wartością atrybutu, krotką wartości lub zbiorem wartości.

Struktura obiektu jest określona przez zbiór nazwanych atrybutów. **Atrybuty** przyjmują wartości z odpowiednich dziedzin, które są zbiorami wartości elementarnych, takich jak liczby naturalne, rzeczywiste czy łańcuchy znaków. Każde wystąpienie obiektu ma taki sam zbiór atrybutów określonego typu.

Jak pamiętamy, konwencjonalne relacyjne bazy danych oferują bardzo ubogi zestaw typów danych, często zredukowany do liczb, łańcuchów znaków i dat. Jest to wystarczające w przypadku tworzenia prostych aplikacji ewidencyjnych, natomiast zdecydowanie niewystarczające w przypadku nowoczesnych dziedzin zastosowań baz danych, takich jak np. komputerowe wspomaganie projektowania. W post-relacyjnych bazach danych istniała możliwość rozszerzenia zestawu predefiniowanych typów danych o nowe typy - **abstrakcyjne typy danych** (ang. *abstract data types (ADT)*). Możliwość definiowania *ADT-ów,* a następnie wskazywania ich przy definicji pól krotek, odpowiada w pewnym stopniu możliwości dynamicznego rozszerzania schematu obiektowej bazy danych o nowe klasy. Pojęcie *ADT-u* wiąże się również z tzw. osłonowaniem (ang. *encapsulation*) z obiektowych baz danych. Zgodnie z nim dostęp do struktury danych odbywa się wyłącznie poprzez operatory (takie jak np. *push, pop*, dla struktury danych zwanej stosem). W ogólności, abstrakcyjny typ danych składa się z definicji struktury oraz definicji operatorów, które mogą być stosowane wyłącznie dla tej struktury.

Relacje posiadają płaską, jednowymiarową strukturę. Utrudnia to modelowanie złożonych encji, takich jak np. samochód lub przedsiębiorstwo. W post-relacyjnych bazach danych istniała możliwość definiowania atrybutów złożonych, które rekurencyjnie mogą składać się ze zbioru innych atrybutów prostych i złożonych. Dostęp do atrybutów zagnieżdżonych odbywa się poprzez pełną specyfikację ścieżki zawierania, tak jak to ma miejsce w przypadku rekordów języka Pascal, lub struktur języka C.

Mechanizm zagnieżdżonych relacji jest szczególnym przypadkiem złożonych struktur danych. W post-relacyjnych bazach danych wartością atrybutu może być relacja, która z użytkowego punktu widzenia jest traktowana podobnie jak relacja zewnętrzna. W szczególności zatem, relacja zagnieżdżona może zawierać atrybuty, których wartościami są kolejne relacje.

Prostym przykładem wykorzystania relacji zagnieżdżonych może być encja *Człowiek*, modelowana jako krotka o atrybutach prostych, takich jak *imię, nazwisko, data urodzenia,* oraz atrybutu *Dzieci* będącego zagnieżdżoną relacją o krotności odpowiadającej liczbie dzieci określonego człowieka.

W systemach post-relacyjnych istniała możliwość definiowania relacji na bazie innych, wcześniej utworzonych relacji przez wskazanie rozszerzeń i/lub różnic. Jest to mechanizm podobny do mechanizmu dziedziczenia obiektowych baz danych.

W ujęciu klasycznym pojęcie dziedziczenia odnosi się przede wszystkim do koncepcji obiektowości w językach programowania i bazach danych. Obiekty programistyczne są grupowane w klasy. Pomiędzy klasami ustanawia się związek hierarchiczny określający zasady dziedziczenia. Przykładowo klasa obiektów *Osoba* posiada podklasy *Pracownik*, *Klient*, *Emeryt*, podklasa *Pracownik* może posiadać podklasy *Pracownik_etatowy*, *Pracownik_na_zlecenie*. Dziedziczenie polega na tym, że pewne wspólne cechy (inwarianty) obiektów należących do klas bardziej ogólnych są dziedziczone przez obiekty należące do klas bardziej szczegółowych. W językach programowania (np. C++ i Smalltalk) takimi inwariantami są definicje atrybutów obiektów (np. fakt występowania i definicja atrybutu *nazwisko* wewnątrz klasy *Osoba* są dziedziczone przez pozostałe wymienione podklasy), oraz tzw. metody, czyli procedury, które można stosować do elementów danej klasy. W obiektowych bazach danych zestaw tych inwariantów ulega rozszerzeniu: dziedziczyć można także wspólne wartości pewnych

atrybutów, wartości domyślne, definicje zdarzeń i reguł aktywnych, definicje więzów ochrony integralności i autoryzacji dostępu, definicje typów, itd. W sytuacji gdy obiekt danej klasy można zaliczyć do dwóch lub więcej klas bardziej ogólnych mamy do czynienia z wielo-dziedziczeniem, np. obiekt *Pracujący_student* dziedziczy inwarianty zarówno z klasy *Pracownik* jak i z klasy *Student*.

Utwórzmy w pewnej przykładowej bazie danych klasę *Pracownik_na_zlecenie.*

create table *Pracownik_na_zlecenie* (nr_zlecenia char16)

inherits (*Pracownik*);

insert into *Pracownik_na_zlecenie* (nr_zlecenia, nazwisko, płaca, wiek, wydział)

values (‘cd-23/93’, ‘A.Kowalski’ , 1200, 23, ‘elektryczny’)

Każde wystąpienie w klasie *Pracownik_na_zlecenie* dziedziczy (ang. *inherits*) wszystkie atrybuty (tj. *nazwisko, placa, wiek, wydział*) z klasy nadrzędnej (inaczej klasy - rodzica) *Pracownik*. Klasa *Pracownik_na_zlecenie* posiada dodatkowo atrybut *nr_zlecenia*. Zapytania mogą dotyczyć danej klasy jak i klasy nadrzędnej. Na przykład, następujące zapytanie

select *

from *Pracownik* ***

where *wiek >65*

spowoduje wyszukanie w klasie *Pracownik* i w *Pracownik_na_zlecenie* wszystkich wystąpień dla których jest spełniony warunek *wiek* > *65*. Gwiazdka * po nazwie klasy *Pracownik* wskazuje, że zapytanie powinno być wykonane na tej klasie i wszystkich klasach znajdujących się niżej w hierarchii dziedziczenia. W ogólności hierarchia dziedziczenia może być acyklicznym grafem skierowanym.

Pojedyncza relacja post-relacyjnej bazy danych może być przechowywana w wielu wersjach. *SZBD* zapamiętuje kolejność tworzenia wersji relacji w postaci drzewa wywodu. W większości propozycji wersje tej samej relacji muszą mieć ten sam schemat - oznacza to, że mogą się różnić wyłącznie zawartością, a nie strukturą. Implementacje tego mechanizmu mają różną postać - w systemie *Postgres* jest to archiwizacja „starych” wartości i zapytania historyczne.

Języki zapytań systemów post-relacyjnych umożliwiały wykonywanie tak zwanych **zapytań historycznych**, to znaczy zapytań dotyczących stanu klasy w określonym punkcie w czasie lub przedziale czasu. Aby można było na klasie wykonywać zapytania historyczne, musi być ona utworzona z archiwizacją starych wartości. Na przykład w systemie *Postgres*, uzyskujemy to przez polecenie *create* z użyciem opcji *archive=light* lub *archive=heavy* (dla aktualizacji przewidywanej rzadko lub często). Do określenia konkretnego momentu w czasie służy typ *abstime*, którego postać jest następująca:

‘miesiąc dzień, rok godzina:min:sek’::abstime

Ponadto można wykorzystywać słowa kluczowe określające szczególne chwile czasu, np.:

- ‘now’, ‘current’ – określa chwilę bieżącą,
- ‘epoch’ – początek zegara systemowego,
- ‘infinity’, ‘-infinity’ – minus nieskończoność, nieskończoność.

Chcąc uzyskać informację o aktualnych zarobkach określonego pracownika można wydać zapytanie w następującej postaci:

select płaca

from *Pracownik* [‘now’]

where nazwisko = ‘Z.Królikowski’

Zauważmy, że użycie specyfikacji ‘now’ jest równoznaczne z pominięciem jej w ogóle. Jeżeli chcemy uzyskać informacje o zarobkach tegoż pracownika w określonym przedziale czasu (np. od 1 stycznia 1982 do chwili obecnej), zapytanie to należy sformułować następująco:

select płaca

from *Pracownik* [‘Jan 1, 1982’, ‘now’]

where nazwisko = ‘Z.Królikowski’

Użytecznym rozszerzeniem post-relacyjnych baz danych była możliwość definiowania atrybutów krotek w postaci procedur zapisanych w języku bazy danych lub jednym z dostępnych w systemie klasycznych języków programowania. Każdorazowe odwołanie się do takiego atrybutu powoduje uaktywnienie związanej z nim procedury i wyliczenie poszukiwanej wartości atrybutu. Przykładem atrybutu proceduralnego może być atrybut *wiek*, który każdorazowo jest wyliczany na bazie atrybutu *data_urodzenia* i pobranej z *SZBD* aktualnej daty. Mechanizm atrybutów proceduralnych w dużym uproszczeniu odpowiada metodom obiektów w obiektowych bazach danych. W systemie *Postgres* mechanizm ten jest realizowany poprzez funkcje języka C i funkcje języka SQL.

Przedstawimy obecnie sposób definiowania atrybutów proceduralnych z wykorzystaniem poleceń języka SQL. Dowolny zestaw poleceń języka SQL może być ujęty w formie pewnej jednostki, wywoływanej i wykonywanej przez podanie jej nazwy i ewentualnie parametrów. Taki zestaw poleceń w systemie *Postgres* nazywa się **funkcją języka zapytań**. (ang. *query language function*).

Przykładowa funkcja *pracownik_dobrze_opłacany*, w postaci przedstawionej poniżej, jako wynik dostarcza wszystkie wystąpienia klasy *Pracownik* dla których spełniony jest warunek:

placa > $1, gdzie *$1* jest nazwą parametru, którego wartość należy podać przy wywołaniu funkcji.

create function *pracownik_dobrze_opłacany* (int4) returns set of *Pracownik*

as ‘select * from *Pracownik* where płaca > $1’

language ‘sql’

Jeśli w definicji funkcji występują parametry, to należy zdefiniować ich typ (dla naszego przykładu - *(int4)*). W definicji funkcji należy również określić język w którym funkcja została zdefiniowana (*language = ‘sql’*) oraz typ wyniku (*returns set of Pracownik* - co tutaj oznacza, że jako wynik wykonania funkcji uzyskamy zbiór wystąpień klasy *Pracownik*, spełniających warunki określone w definicji funkcji, tj. po słowie kluczowym *where*).

Omówione powyżej funkcje można wykorzystać do budowania tak zwanych **atrybutów wirtualnych**. Rozważmy możliwość rozszerzenia klasy *Pracownik* o atrybut *kierownik*. To znaczy z każdym wystąpieniem w klasie *Pracownik*, chcemy związać inne wystąpienie z tej klasy określające kierownika danego pracownika. W tym celu należy zdefiniować następującą funkcję:

create function *kierownik* (*Pracownik*) returns *Pracownik*

as ‘select P. * from *Pracownik* P., *Wydziały* W

where P.nazwisko = W.kierownik and $1.wydział = W.nazwa’

language ‘sql’;

Wykonanie funkcji *kierownik* możemy traktować jako dodanie do klasy *Pracownik* atrybutu wirtualnego złożonego typu *kierownik*, którego wartością jest wystąpienie klasy *Pracownik*. Innymi słowy, funkcję *kierownik* użytkownik może traktować jak zwyczajny atrybut w klasie *Pracownik* i wykonywać na nim dowolne operacje, np.

select *Pracownik*.nazwisko

where nazwisko(kierownik(*Pracownik*)) = '*Z.Królikowski'*;

to znaczy, "wyszukaj wszystkich pracowników, których kierownikiem jest *Z.Królikowski"*.

Rozszerzenia języka zapytań *SZBD*, takie jak omówione powyżej na przykładzie systemu *Postgres*, pojawiają się już również w najnowszych wersjach systemów komercyjnych, jak na przykład w systemie *Ingres*, *Oracle, DB2, Sybase i Informix*. W tym kontekście w systemie *Ingres* operuje się właśnie pojęciem **atrybutów wirtualnych** tj. atrybutów, których wartość jest obliczana w momencie wykonywania operacji dostępu.

4. Systemy aktywnych baz danych

Poniżej przedstawiono genezę i krótką charakterystykę systemów aktywnych baz danych. Tej klasie systemów w ramach naszego kursu będzie poświęcony osobny wykład.

Jeszcze kilkanaście lat temu, większość systemów baz danych były to systemy pasywne, w których wszystkie operacje manipulowania danymi były związane tylko i wyłącznie z realizacją żądań użytkowników. Natomiast najogólniej mówiąc, **systemy aktywnych bazy danych** są systemami, które „żyją" również wtedy, kiedy nie są do nich jawnie kierowane żadne transakcje (żądania użytkowników). Oznacza to, że w bazach tych istnieje możliwość zmian stanu ich krotek (obiektów) również poza sesjami użytkowników, np. na skutek zajścia określonego zdarzenia zewnętrznego. Poza zdarzeniami zewnętrznymi, „uaktywnienie" bazy danych może nastąpić na przykład w następujących sytuacjach:

- na skutek zakończenia realizacji określonego zbioru transakcji kierowanych do *SZBD* przez jej użytkowników, np. po wprowadzeniu ocen egzaminacyjnych z sesji studenta jest automatycznie uruchamiana transakcja obliczająca średnią jego ocen i wysokość przysługującego mu stypendium naukowego;
- na skutek upływu określonego kwantu czasu, np. co każdy dzień, bez ingerencji użytkownika uruchamiane są określone transakcje lub programy aplikacyjne;
- kombinacji dwóch powyższych przypadków, np. po realizacji czeku powodującego debet na koncie jest realizowana blokada konta i cyklicznie, co jeden dzień, są naliczane odsetki karne.

W odróżnieniu od tradycyjnych systemów pasywnych systemy aktywnych baz danych są wyposażone w mechanizmy (odpowiedni język) definiowania i przetwarzania **reguł aktywnych *Event-Condition-Action (ECA)*** (ang. *active rules*). Poniżej przedstawiono krótką charakterystykę tych reguł. Ich zdefiniowanie w ramach badań nad post-relacyjnymi bazami danych, stanowiło podstawę rozwiązań stosowanych w systemach komercyjnych. We współczesnych komercyjnych SZBD jest to mechanizm wyzwalaczy (ang. *triggers*).

Logika reguł *ECA* (inspirowana regułami produkcyjnymi z systemów eksperckich) jest następująca: w wyniku zajścia zdarzenia *E*, jeśli spełniony jest warunek *C*, podejmowana jest akcja *A*. Reguły aktywne są przechowywane w bazie danych i mogą być współdzielone przez różne programy aplikacyjne. Reguły aktywne stanowią spójny mechanizm umożliwiający z poziomu systemu zarządzania bazą danych wykonywanie zmian stanu bazy danych, egzekwowanie więzów integralności i praw dostępu, monitorowanie dostępu do danych, śledzenie ewolucji bazy danych oraz otrzymywanie danych pochodnych.

Korzenie języków regułowych baz danych wywodzą się ze sztucznej inteligencji. Reguły definiowane w ramach tej dziedziny wiedzy noszą nazwę reguł produkcyjnych i przyjmują następującą postać:

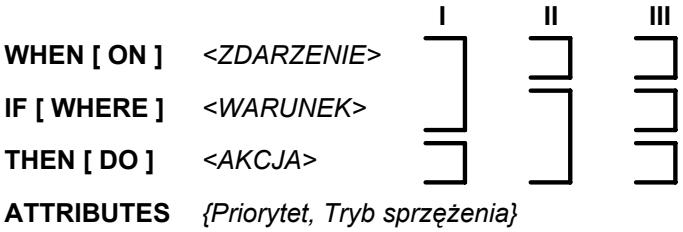
wzorzec → akcja

Reguła jest uaktywniana gdy **wzorzec** (ang. *pattern*) "pasuje" do danych znajdujących się w przestrzeni pamięci roboczej systemu, natomiast **akcja** (ang. *action*) dokonuje modyfikacji

pamięci roboczej, zgodnie z dopasowaniem danych. Wzorzec jest również określany mianem warunku (ang. *condition*) lub predykatu (ang. *predicate*). Innymi słowy, przyjmuje się założenie, że reguła jest uaktywniana tylko wtedy gdy w pamięci roboczej pojawią się nowe dane pasujące do wzorca (lub w przypadku reguł zanegowanych, gdy pasujące dane są usuwane z pamięci roboczej). Tak więc reguły pośrednio są uaktywniane przez zdarzenia takie jak wprowadzenie nowych danych oraz modyfikacje lub usunięcie danych.

Podstawowa różnica pomiędzy językami regułowymi sztucznej inteligencji a językami stosowanymi w systemach baz danych polega na tym, że **zdarzenia** (ang. *events*) uaktywniające reguły są specyfikowane w sposób bezpośredni.

Języki reguł aktywnych implementowane w różnych systemach aktywnych baz danych różnią się znacznie biorąc pod uwagę ich semantykę, syntaktykę oraz wykorzystywane mechanizmy wnioskowania. Reguły w systemach aktywnych baz danych mogą przyjmować postać jak na rysunku 3.



Rys. 3. Możliwe postacie reguł aktywnych

Jak na razie został osiągnięty kompromis dotyczący elementarnych składników reguł aktywnych - powinny się one składać z wyrażeń definiujących: zdarzenia, warunki, akcje oraz zestawu atrybutów. Reguła jest uaktywniana gdy wystąpi określone zdarzenie. Warunek zdefiniowany w treści reguły jest sprawdzany na podstawie danych. Jeśli warunek jest spełniony podejmowana jest określona akcja. Powyższy sposób reprezentacji reguł stał się standardem znanym jako reguły *ECA*. Jak wynika z przedstawionego powyżej formatu reguły nie osiągnięto jednak konsensusu dotyczącego nazewnictwa (słów kluczowych występujących w definicji reguły). Również szczegóły implementacyjne i stopień złożoności specyfikacji zdarzeń, warunków i akcji różnią się znacznie w poszczególnych systemach.

Przypadek I z rys. 3 powyżej dotyczy przede wszystkim systemów z regułami produkcyjnymi (np. *OPS5*). W niektórych systemach komercyjnych takich jak *Sybase*, *Interbase*, wykorzystywano początkowo definicję II reguł aktywnych. Wówczas akcja jest transakcją, natomiast część warunkowa jest zakodowana jako jej część składowa. W systemach eksperymentalnych rozszerzalnych i aktywnych systemów baz danych, takich jak: *Postgres*, *Starburst*, *HiPAC* wykorzystuje się definicję III.

W niektórych językach dane związane ze zdarzeniem i/lub częścią warunkową mogą być przekazywane do części warunkowej i/lub do definicji akcji. Pewne języki (np. w systemie *Starburst*) są wyposażone w mechanizmy porządkowania reguł (ang. *rule ordering*), których celem jest określenie, jaka reguła powinna być wykonana jeśli wiele reguł zostało uaktywnionych (ang. *conflict resolution*).

5. Systemy mobilnych baz danych

Systemy mobilnych baz danych nie będą szczegółowo omawiane w ramach naszego kursu. Poniżej przedstawiono ich krótką charakterystykę oraz wskazano listę problemów badawczych i implementacyjnych, z których większość ma nadal charakter otwarty. Tym z Państwa, którzy są zainteresowani systemami mobilnymi, proponuje zapoznanie się z materiałami dydaktycznymi opracowanymi w ramach kursu „Systemy mobilne”.

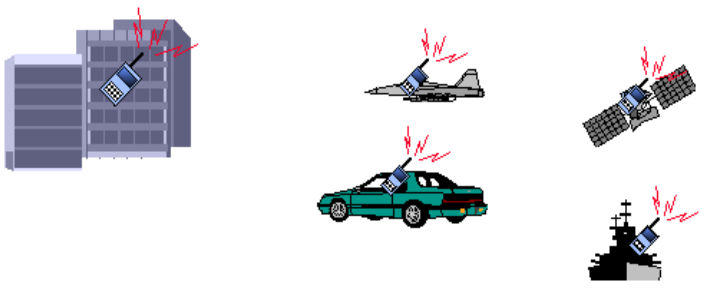
Na początek kilka słów wprowadzenia w problematykę systemów mobilnych. W czasach obecnych utarło się przekonanie, iż żyjemy w społeczeństwach informacyjnych. Niektórzy twierdzą, że ten kto ma informację ma władzę, inni uważają, że ma ona wartość złota. U schyłku XX wieku, kiedy świat stał się „globalną wioską" szybki dostęp do danych jest kluczowy w wielu dziedzinach życia. Natychmiastowe uzyskanie informacji na dany temat stało się sprawą priorytetową dla prowadzonych interesów czy synchronizacji pracy w firmie. Jest ono również ważne w kwestiach dotyczących bezpieczeństwa obywateli nie wyłączając z tego zastosowań wojskowych. Mając pełniejszy obraz sytuacji możemy robić te same rzeczy lepiej, dokładniej i prędzej. Możemy wyprzedzić tych, którzy dostępu takowego nie posiadają.

Komputery znacznie usprawniły proces gromadzenia i przetwarzania informacji. Korzystając z zasobów danych zgromadzonych na całym świecie, mając dostęp do relatywnie szybkich mediów transmisyjnych jak i całej gamy usług, możemy uzyskać potrzebne nam informacje w sposób szybki i pewny. Pojawia się jednak jeden zasadniczy problem. Czy każdy użytkownik, o każdej porze i niezależnie od miejsca, gdzie się teraz znajduje może uzyskać dostęp do potrzebnych mu danych? Czy podróżując lub będąc w miejscu, gdzie nie ma żadnej tradycyjnej infrastruktury sieciowej, możemy korzystać ze wszystkich dostępnych zasobów. Czy na obecnym poziomie technologicznym istnieją możliwości zapewnienia użytkownikowi ruchomemu wszelkich serwisów oferowanych przez tradycyjne systemy?

Odpowiedź na postawione wyżej pytanie brzmi: ***tak***. Postęp technologiczny w zakresie bezprzewodowych sieci LAN oraz satelitarnych usług komunikacyjnych spowodował, iż stał się możliwy dostęp do informacji przez użytkowników mobilnych, czyli będących w ruchu. Korzystając z bezprzewodowych połączeń użytkownik mobilny może korzystać z wszelkich danych w dowolnym czasie i niezależnie od miejsca, gdzie się znajduje. Już obecnie miliony ludzi, używając przenośnych komputerów, podróżuje po całym świecie produkując i konsumując informacje.

Z budową i rozwojem **systemów mobilnych baz danych** wiąże się wiele pytań i problemów zarówno sprzętowych jak i programowych. Mobilność użytkowników powoduje, iż nie wiemy gdzie znajduje się adresat danych. Nasuwa się myśl jak w sposób maksymalnie efektywny odnaleźć użytkownika, do którego skierowany jest komunikat. Czy możemy mieć pewność, że mobilny komputer aktualnie nie został wyłączony w celu oszczędności źródeł zasilania. Czy dane po przywróceniu łączności do użytkownika dotrą ? Rodzi się też pytanie jak odpowiadać na zapytania, które są zależne od pozycji użytkownika. Czy dane przesyłać na żądanie klienta, czy periodycznie wysyłać je w eter ? Te i mnóstwo innych problemów stają przed specjalistami w tej dziedzinie, pozostawiając wiele spraw i problemów otwartymi.

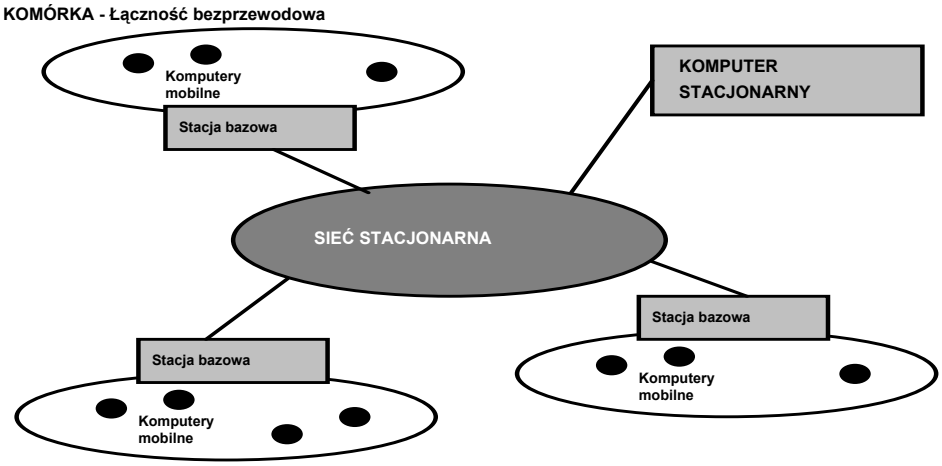
Próba znalezienia odpowiedzi na postawione wyżej pytania i propozycje rozwiązań technologicznych są prezentowane w ramach kursu „Systemy mobilne". Tutaj przedstawimy tylko pewne definicje i koncepcje wstępne dotyczące systemów mobilnych baz danych.



Rys. 4. Użytkownicy mobilni

W systemach mobilnych baz danych, słowo **mobilny** definiuje się jako „zdolny do ruchu w czasie przyłączenia do sieci". Można zatem powiedzieć, że użytkownik mobilny jest to użytkownik (najczęściej będący w ruchu), korzystający z komputera mobilnego zdolnego do

przyłączenia się do tradycyjnej sieci za pomocą połączenia bezprzewodowego. Jak widać użytkownik mobilny nie zawsze musi być związany z ruchem (jest on do zdolny do ruchu).



Rys. 5. Elementy składowe systemu mobilnego

Środowisko mobilne składa się z elementów stałych i ruchomych. Komputery stacjonarne, będące elementami stałymi systemu mobilnego, połączone są tradycyjną siecią zapewniającą duże przepustowości (rys. 5). Nie mogą one jednak bezpośrednio łączyć się z użytkownikami mobilnymi, wobec tego w sieć tą włączono stacje bazowe odpowiedzialne za łączność z nimi. Są one interfejsem pomiędzy komputerami stacjonarnymi a użytkownikami w terenie. Każda stacja bazowa pokrywa swoim zasięgiem obszar zwany komórką.

Użytkownik mobilny znajdujący się w danej komórce komunikuje się tylko z jedną stacją bazową, odpowiadającą za komórkę, w której aktualnie przebywa. Mówimy wtedy, że taki komputer mobilny jest lokalny dla danej stacji bazowej. Lokalizację komputera mobilnego definiuje się właśnie jako komórkę, w której się on znajduje. Jako ruch użytkowników mobilnych definiuje się ich migrację pomiędzy komórkami.

Jak więc widać w systemie mobilnym połączeniami bezprzewodowymi są jedynie kanały stacja bazowa - komputer mobilny. Stacje bazowe oraz serwery stacjonarne komunikują się korzystając z tradycyjnych sieci komputerowych zapewniających duże przepustowości.

6. Systemy multimedialnych baz danych

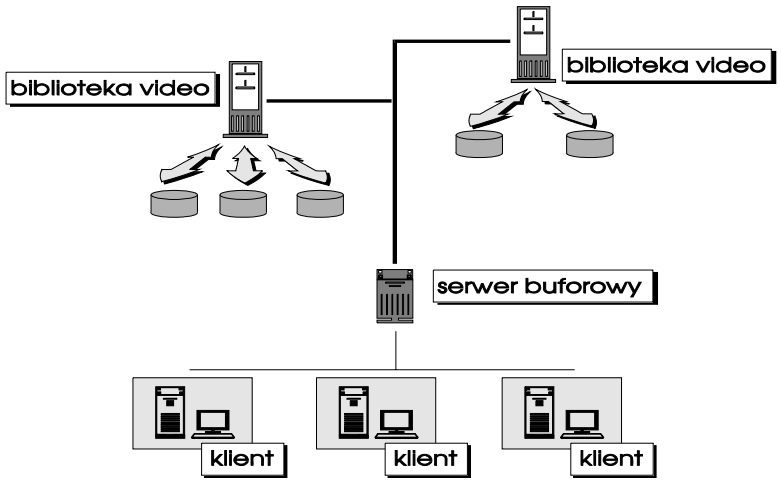
Poniżej przedstawiono genezę i krótką charakterystykę systemów multimedialnych baz danych. Tej klasie systemów w ramach naszego kursu będą poświęcone dwa wykłady.

Nieustanna poprawa jakości parametrów technicznych urządzeń pamięci masowej, sieci komputerowych, urządzeń wejścia-wyjścia i elementów obliczeniowych pozwoliła przekroczyć kolejne granice zastosowań systemów baz danych. Proste typy i struktury danych stosowane w systemach tradycyjnych mogą zostać uzupełnione przez cyfrowy zapis obrazu, dźwięku, video i mowy ludzkiej. Wymagania sprzętowe dla **systemów multimedialnych baz danych** są na tyle wysokie, że możliwość praktycznego zastosowania komputerów do przetwarzania i odtwarzania informacji multimedialnych pojawiła się dopiero niedawno.

Pojęcie **multimedia** definiuje się jako połączenie konwencjonalnych typów danych z tekstem, obrazem, grafiką i animacją komputerową, dźwiękiem, mową ludzką i video. Nowe typy danych stanowią rozszerzenie typów tradycyjnych w kierunku dodatkowych wymiarów (geometrycznych i czasu). W związku z pojawieniem się aspektu temporalnego w danych multimedialnych klasyfikuje się je jako *ciągłe* lub *dyskretne*. Przy prezentacji mediów ciągłych (dźwięk, mowa, animacja, video) musi być uwzględniany wymiar czasu. Treść mediów dyskretnych (tekst,

obraz, grafika komputerowa, typy konwencjonalne) nie jest czasowo uwarunkowana, stąd wymiar czasu nie musi być uwzględniany podczas prezentacji.

Pierwszą fizycznie dostrzegalną cechą cyfrowych postaci nowych mediów jest ich znaczny rozmiar w porównaniu z pojemnościami powszechnie dostępnych pamięci operacyjnych i masowych. Jest to wyraźnie widoczne w przykładzie z biblioteką video na żądanie (rys. 6).



Rys. 6. Systemy multimedialnych baz danych – biblioteka video na żądanie

Biblioteka taka składa się z centralnego komputera, w którym są pamiętane (w postaci cyfrowej) filmy oraz przyłączy do sieci komputerowej. Sieć łączy komputer centralny (serwer) z komputerami w domach abonentów. Załóżmy, że do sieci jest dołączonych 1000 abonentów i że średnie wykorzystanie sieci to 50%, to znaczy z sieci w danym czasie korzysta połowa abonentów. Biblioteka zawiera 500 filmów, każdy trwający 1.5 godziny. Każdy abonent jest obsługiwany niezależnie - może w dowolnej chwili zatrzymać oglądany film, cofnąć lub przeglądać na przyspieszonym podglądzie, zatrzymać i wyświetlać pojedynczą klatkę filmu itp.

Wymagania sprzętowe dla serwera, przy założeniu kodowania cyfrowego w trybie *MPEG*, to:

500 filmów $\times$ 5400 sekund $\times$ 1,5 Mb/sek. = ok. 4 Tb = ok. 500 GB.

Przy założeniu, że są wykorzystywane obecnie produkowane napędy dyskowe, oznacza to konieczność dołączenia do serwera ok. 10 napędów dyskowych o pojemności 50GB każdy. Jest to obecnie bez problemu fizycznie wykonalne.

Przy założeniu, że jednocześnie 50% abonentów może korzystać z biblioteki, szybkość transmisji w głównym przyłączu sieciowym serwera musi wynosić:

500 abonentów $\times$ 1.5 Mb/sek. = 750 Mb/sek. = 100MB/sek.

Szybkość ta jest porównywalna z szybkością transmisji w sieciach FDDI i ATM. Szybkość transmisji dyskowych (przy założeniu optymalnego rozłożenia żądań abonentów między napędy dyskowe) musi wynosić co najmniej:

100 MB/sek. / 50stacji dysków = 2 MB/sek.

Od strony klienta jest wymagany komputer klasy PC, z wysokorozdzielczym ekranem kolorowym (lub przyłączem do odbiornika telewizyjnego) oraz odpowiednią kartą dźwiękową i kartą sieciową. Ponieważ szybkość transmisji sygnału wideo nie przekracza 1.5 Mb/sek., zupełnie wystarczająca jest zwykła karta sieciowa (Ethernet lub Token Ring) o szybkości transmisji 10 Mb/sek., przy założeniu topologii sieci typu gniazda lub podziale abonentów na segmenty sieciowe o wielkości 5÷6 przyłączy.

Jest zatem widoczne, że od strony sprzętowej już kilka lat temu został osiągnięty poziom, który umożliwia przechowywanie, przesyłanie i odtwarzanie informacji o charakterze multimedialnym.

Inne cechy charakterystyczne danych multimedialnych są następujące. Nie ma ogólnie uzgodnionego standardu przechowywania informacji multimedialnej, tak jak to ma miejsce w przypadku muzyki i płyt CD lub wideo i standardu VHS. Istnieje wiele urządzeń do przechowywania informacji multimedialnej w formie cyfrowej, ale praktycznie każde z nich przechowuje informację w innym standardzie (choć w ostatnich latach sytuacja w tym zakresie ulega zmianie). Także sposób przesyłania danych multimedialnych nie jest ściśle ujednolicony. Z reguły nie pozwala to na łatwą integrację całości systemu. Z drugiej strony, stosowanie w jednym systemie wielu specjalizowanych urządzeń jest często uzasadnione ich szczególnymi parametrami - na przykład oferowaną szybkością transmisji lub pojemnością.

W przeciwieństwie do klasycznych danych, informacje multimedialne charakteryzują się bardzo silnymi uwarunkowaniami czasowymi. Na przykład, informacje wideo muszą być przekazywane jako lista ramek, wyświetlanych jedna po drugiej na tyle szybko, aby ludzkie oko wychwytywało je jako obraz ciągły. Tak więc, film można traktować jako sekwencję pojawiających się kolejno obrazów w połączeniu z towarzyszącym mu dźwiękiem. Z przekazem filmu wiąże się wiele problemów natury technicznej. Do podstawowych należą bardzo duża szybkość transmisji sekwencji wideo i bardzo duża zajętość pamięci. Są stosowane różnorodne metody kompresji sygnału wizyjnego. Im lepszy współczynnik kompresji, tym większe wymagania na moc obliczeniową procesora - dekompresja musi być wykonana w czasie rzeczywistym. W ogólności, przechowywanie danych multimedialnych wymaga zakodowania także informacji o czasie i sposobie ich prezentacji.

W przypadku prezentacji multimedialnych jest konieczne zapewnienie synchronizacji w odtwarzaniu różnych elementów przekazu (na przykład obrazu i dźwięku filmu). Razem z danymi należy więc pamiętać (i przetwarzać w czasie rzeczywistym) także informację synchronizacyjną.

Zasygnalizowane powyżej problemy to tylko mała część ważnych zagadnień, które należy rozwiązać, aby koncepcja multimedialnych baz danych mogła być szeroko zaimplementowana w praktyce. Pozostałe z tych zagadnień zostaną przedstawione w ramach wspomnianych powyżej dwóch wykładów poświęconych systemom multimedialnych baz danych.

7. Magazyny (hurtownie) danych

Informatyzacja firm, instytucji i innych jednostek organizacyjnych powinna realizować dwa podstawowe cele. Po pierwsze, powinna usprawnić pracę pojedynczego pracownika: sprzedawcy, magazyniera, księgowego lub urzędnika, poprzez automatyzację realizowanych przez nich wybranych, rutynowych działań. Przykładem takich działań, które nadają się do automatyzacji są: wprowadzanie zamówień, wydawanie lub przyjmowanie towaru, realizacja sprzedaży, rezerwacja miejsc lub operacja przelewu na kontach bankowych. Działania te charakteryzuje ściśle określona procedura postępowania i cykliczna powtarzalność. Dzięki automatyzacji działania te powinny być wykonywane szybciej i w sposób bardziej niezawodny.

Drugim celem informatyzacji jest racjonalizacja działania całych firm, w wyniku wspomagania decyzji kadry zarządzającej przez dostarczenie danych analitycznych opisujących bieżący stan i historię działania danej firmy. Programowe narzędzia analityczne, na podstawie danych elementarnych będących wynikiem pracy pojedynczych pracowników, powinny udostępniać informacje statystyczne o bieżącym stanie firmy, występujących trendach i korelacjach między różnymi czynnikami. Dzięki szybkiej analizie bazującej na pełnej i aktualnej informacji o stanie firmy, kadra zarządzająca może podejmować trafniejsze decyzje o strategicznym znaczeniu dla rozwoju danego przedsiębiorstwa.

Aplikacje operacyjne systemu informatycznego, których celem jest wspomaganie pracy pojedynczych pracowników charakteryzują się prostym przetwarzaniem. Są ograniczone zazwyczaj do niewielkiego zbioru danych szczegółowych i realizują proste operacje odczytu, wstawiania, modyfikacji i usuwania danych. Modelem przetwarzania właściwym dla tej kategorii aplikacji jest tak zwane **przetwarzanie transakcyjne** (ang. *On-line Transaction Processing* -

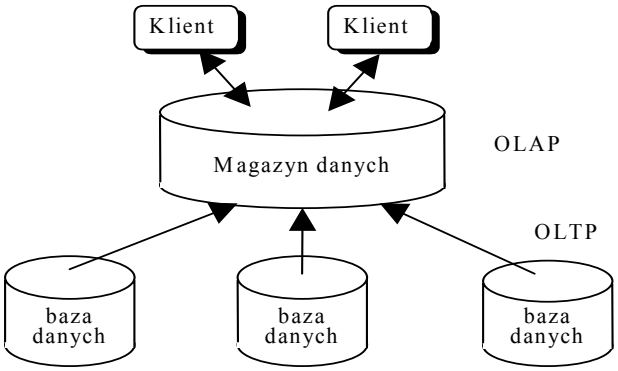
OLTP). Opiera się on na pojęciu elementarnej jednostki przetwarzania danych, tak zwanej *transakcji*. Transakcja charakteryzuje się własnościami: niepodzielności, spójności, izolacji i trwałości (ang. *Atomicity*, *Consistency*, *Isolation*, *Durability* - *ACID*), które gwarantują spójne przetwarzanie danych. Głównym celem technologii związanej z przetwarzaniem transakcyjnym jest zapewnienie spójności danych oraz wysoka wydajność systemów informatycznych pracujących w środowisku wielodostępnym - krytycznym parametrem efektywnościowym takich systemów jest ich przepustowość, mierzona liczbą transakcji w jednostce czasu.

Aplikacje analityczne systemu informatycznego, których celem jest wspomaganie pracy kadry zarządzającej cechuje dużo większa złożoność przetwarzania. Aplikacje analityczne nie są przeznaczone do przetwarzania szczegółowych i aktualnych danych w krótkich transakcjach, ale są zorientowane na wspieranie procesów decyzyjnych czyli przetwarzanie danych historycznych, zagregowanych i często skonsolidowanych z kilku źródłowych baz danych. Większość operacji wykonywanych przez tego typu systemy to złożone zapytania wymagające dostępu do milionów krotek, wymagające wielu operacji połączenia i agregowania. Przykładami takich zapytań mogą być: Jaki jest trend sprzedaży towarów z branży AGD w ostatnich kilku tygodniach ? Jaki jest rozkład sprzedaży lodówek w województwie krakowskim ? Tak więc, przetwarzanie w aplikacjach analitycznych charakteryzuje się operacjami odczytu dużych wolumenów danych, przetwarzanych następnie przez złożone funkcje analityczne, oraz operacjami inteligentnego wyszukiwania danych, czyli tak zwanej **eksploracji danych** (ang. *data mining*). Modelem przetwarzania właściwym dla tej kategorii aplikacji jest **przetwarzanie analityczne** (ang. *On-line Analytical Processing* - *OLAP*). Efektywność takich systemów nie jest już mierzona przepustowością transakcji, gdyż taki parametr jak czas odpowiedzi, ma dużo większe znaczenie.

Implementacja systemów informatycznych wspomagających przetwarzanie analityczne wymaga rozwiązania dwóch podstawowych problemów: integracji heterogenicznych i rozproszonych systemów informatycznych oraz efektywności przetwarzania analitycznego w trybie on-line. Dotychczasowe próby rozwiązania obydwu powyższych problemów były przeprowadzane w dużej części niezależnie od siebie. Z jednej strony poszukiwano sposobów integracji heterogenicznych i rozproszonych systemów baz danych, charakteryzujących się przetwarzaniem transakcyjnym. Z drugiej poszukiwano architektury i technologii właściwej dla wydajnego przetwarzania analitycznego.

Prace na środowiskiem dla efektywnego przetwarzania analitycznego zaowocowały technologią magazynów danych. **Magazyny danych**, są to „*tematycznie zorientowanymi, zintegrowanymi, zmiennymi w czasie, nie ulotnymi zbiorami danych, wykorzystywanymi w organizacjach głównie do podejmowania decyzji*”. Magazyny danych są implementowane jako ogromne bazy danych, niezależne od operacyjnych baz danych, na których działają aplikacje *OLTP*. Powodów, dla których systemy *OLAP* nie mogą wykorzystywać operacyjnych baz danych jest kilka. Po pierwsze, procesy decyzyjne wymagają danych, na przykład o trendach, których może nie być w operacyjnych bazach danych. Po drugie, procesy decyzyjne wymagają dostępu do skonsolidowanych danych pochodzących z wielu heterogenicznych źródeł, które mogą używać niezgodnych formatów danych i niezgodnego kodowania. Po trzecie, operacje typowe dla systemów *OLAP* wymagają specjalnego składowania danych, odpowiednich struktur i metod dostępu do danych, których nie stosuje się w tradycyjnych, komercyjnych systemach zarządzania bazami danych.

Tak więc, magazyn danych jest wydzieloną bazą danych (rys. 7) nastawioną na przetwarzanie analityczne i zbiorem narzędzi analitycznych. Architektura magazynu danych zakłada pełną izolację przetwarzania operacyjnego i analitycznego. Informacje powstające w systemach operacyjnych baz danych o przetwarzaniu transakcyjnym są replikowane i magazynowane na zapas dla późniejszego przetwarzania analitycznego.



Rys. 7. Magazyn danych - rozdzielenie przetwarzania operacyjnego i analitycznego

W magazynie danych przechowywane są 4 podstawowe kategorie danych:

- dane elementarne pozyskane bezpośrednio ze źródłowych baz danych;
- dane historyczne tworzone w momencie pojawiania się nowych wartości już przechowywanych danych;
- dane sumaryczne o różnym stopniu przetworzenia;
- dane opisujące semantykę, pochodzenie i algorytmy wyznaczania poprzednich trzech typów danych.

W relacyjnych bazach danych, dane są zorganizowane w relacjach, z których każda może zawierać pewien zbiór krotek. Wszystkie krotki relacji mają tą samą strukturę, czyli ten sam zbiór atrybutów. Jeden lub kilka atrybutów tworzy klucz podstawowy relacji służący do identyfikowania krotek. Jeśli relacja jest odpowiednio znormalizowana to każda krotka opisuje pojedynczy obiekt lub fakt ze świata rzeczywistego. Przykładowa relacja została pokazana na rys. 8.

Nawigując (hipotetycznie) po krotkach takiej relacji, poruszamy się wzdłuż tylko jednego wymiaru - wymiaru obiektów lub faktów, o których informacje są przechowywane w tej relacji. Zbiór identyfikatorów klientów, możemy traktować jako punkty na osi współrzędnych, po której możemy się poruszać, aby odszukać informację o kliencie. Jeśli chcemy odpowiedzieć na pytanie, jaki jest adres danego klienta, musimy znać jego identyfikator będący współrzędną, pod którą można go znaleźć.

Nazwa klienta	*Adres klienta*	*Telefon*
Alfa	ul. Akacjowa 4	8345-543
Beta	ul. Konwaliowa 8	8665-545
Gamma	ul. Klonowa 34/36	8434-221
Delta	ul. Albańska 8	8665-645

Rys. 8. Przykładowa relacja - jeden wymiar.

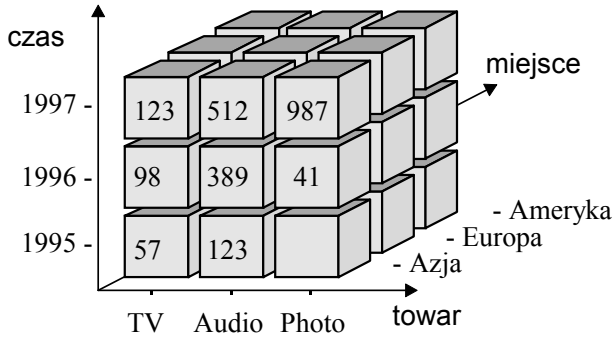
W wielu przypadkach, szczególnie przy przetwarzaniu analitycznym, aby udzielić odpowiedzi na zapytanie, musimy znać więcej niż jedną współrzędną. Na przykład, jeśli w bazie danych posiadamy oprócz klientów informacje o towarach i sprzedaży, możemy być zainteresowani, ile towaru X sprzedano klientowi Y. Mamy zatem dwa niezależne wymiary: wymiar klientów i towarów, a na przecięciu informację o sprzedaży. Tego typu wiedzę w naturalny sposób można reprezentować w postaci struktury dwuwymiarowej pokazanej na rys. 9.

Klient	Towar: Lodówka	Pralka	Zmywarka
Alfa	20	23	5
Beta	4	0	24
Gamma	45	147	35
Delta	71	12	40

Rys. 9. Sprzedaż pokazana w dwóch wymiarach: towary i klienci.

Wiersze takiej tabeli nie mogą być przechowywane w bazie danych jako krotki relacji, ponieważ kolumny tabeli mają charakter całkowicie dynamiczny. Konieczne jest zastosowanie wielowymiarowych struktur danych (rys. 10).

Korzyści wynikające ze stosowanie struktur wielowymiarowych w magazynach danych do przechowywania informacji nie ograniczają się tylko do przejrzystego reprezentowania wiedzy, ale mają przede wszystkim znaczenie efektywnościowe.



Rys. 10. Przykład danych wielowymiarowych.

Na rys. 10 pokazano przykład danej wielowymiarowej obrazującej sprzedaż towarów elektronicznych przez hipotetyczną firmę handlową. Dana ta ma trzy wymiary: czas, typ towaru i lokalizacja sprzedaży. Sześciany reprezentują poszczególne komórki danej wielowymiarowej. Miarą danej wielowymiarowej jest wielkość sprzedaży. Na rysunku wartości miary są reprezentowane przez liczby skojarzone z poszczególnymi komórkami. Dla niektórych komórek, miara jest nieokreślona. Przykładem jest brak wartości miary dla sprzedaży sprzętu fotograficznego w roku 1995 na terenie Azji.

Problematyka magazynów danych jest bardzo szeroka i złożona. W przedstawionej powyżej krótkiej prezentacji tej grupy systemów zaawansowanych baz danych, pominięto szereg problemów związanych z architekturą i projektowaniem magazynów danych oraz możliwościami i efektywnością przetwarzania analitycznego. Zostaną one przedstawione szczegółowo w ramach 2 ostatnich wykładów wchodzących w skład tego kursu.

8. Podsumowanie

Powyżej przedstawiono krótką charakterystykę wybranych systemów zaliczanych do generacji zaawansowanych systemów baz danych. Ograniczono się tylko do zasygnalizowania ważnych problemów naukowych i implementacyjnych. Mam nadzieję, że zasygnalizowane tutaj problemy zachęcą Państwa do podjęcia samodzielnych, bardziej szczegółowych studiów z dziedziny

zaawansowanych systemów baz danych. Pełna prezentacja tych problemów i wyczerpująca charakterystyka systemów:

- rozproszonych baz danych
- aktywnych baz danych,
- obiektowych i relacyjno-obiektowych baz danych,
- multimedialnych baz danych,
- semistrukturalnych baz danych oraz
- magazynów danych,

zostanie przedstawiona w ramach kolejnych wykładów z cyklu poświęconego zaawansowanym systemom baz danych. Życzę Państwu owocnych studiów.