Wprowadzenie

- Zastosowania technologii baz danych
- Kluczowe definicje
- Podejście oparte o pliki vs bazodanowe
- Elementy systemu baz danych
- Zalety systemów bazodanowych i SZBD

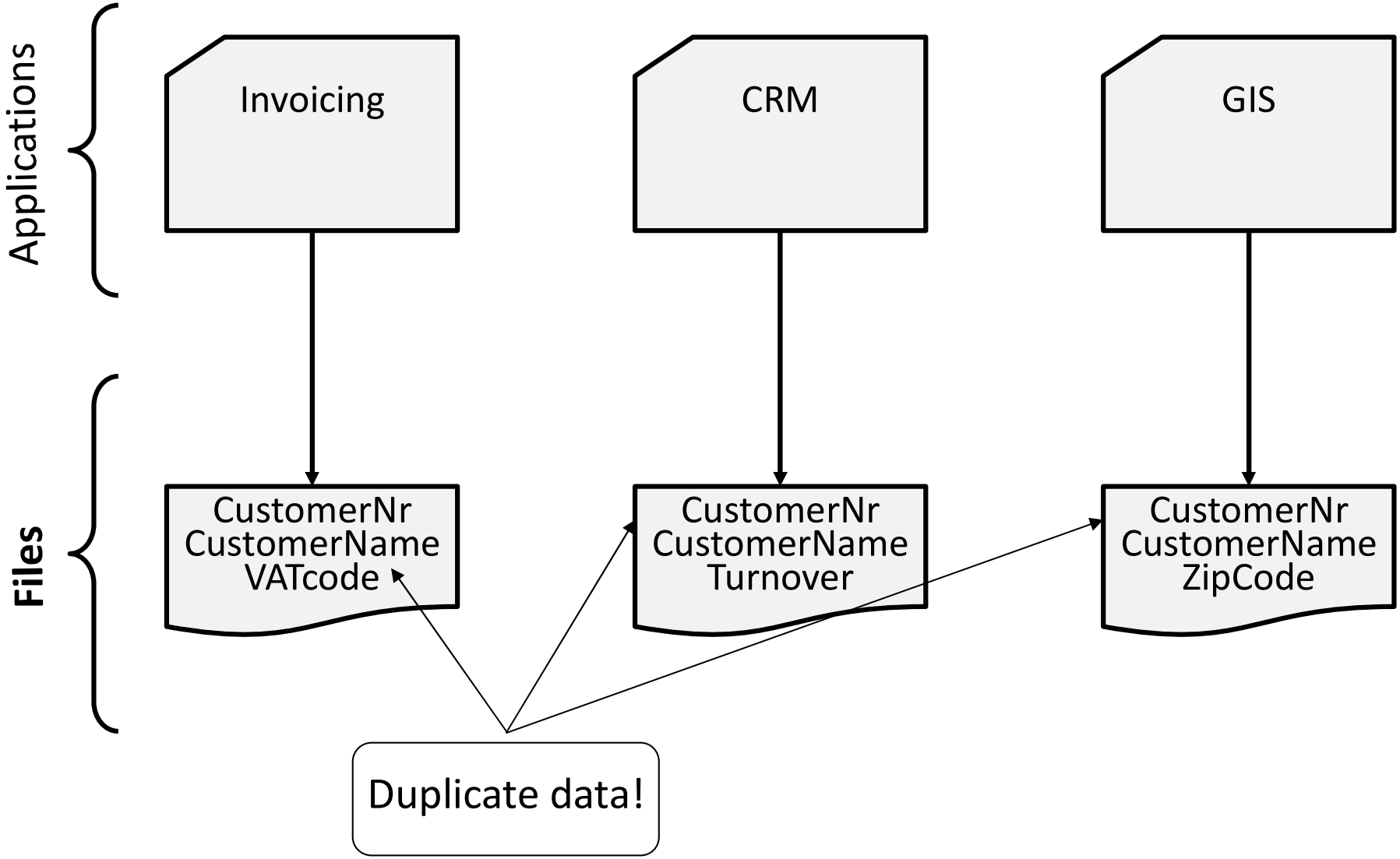
Zastosowania technologii baz danych

1. Przechowywanie i wyszukiwanie tradycyjnych danych numerycznych/alfanumerycznych (śledzenie stanów magazynowych/zamówień/sprzedaży/dostaw)
2. Aplikacje multimedialne (YouTube, Spotify…)
3. Aplikacje biometryczne (odciski palców, skany siatkówki…)
4. Aplikacje mobilne (FitBit, Apple Watch…)
5. Systemy Informacji Geograficznej (GIS) (Google Maps…)
6. Czujniki (reaktor jądrowy…)
7. Aplikacje Big Data (Walmart…
8. Aplikacje Internetu Rzeczy (IoT) (Telematics, hulajnogi..)

Kluczowe definicje

- *Baza danych* – kolekcja powiązanych elementów danych w ramach określonego procesu biznesowego lub podstawionego problemu
  - ma docelową grupę użytkowników i aplikacji
- *System Zarządzania Bazami Danych* (DBMS) - pakiet oprogramowania używany do definiowania, tworzenia, używania i utrzymywania bazy danych
  - składa się z kilku modułów oprogramowania
- Połączenie DBMS i bazy danych jest często nazywane *systemem bazodanowym*

Podejście oparte o pliki vs bazodanowe

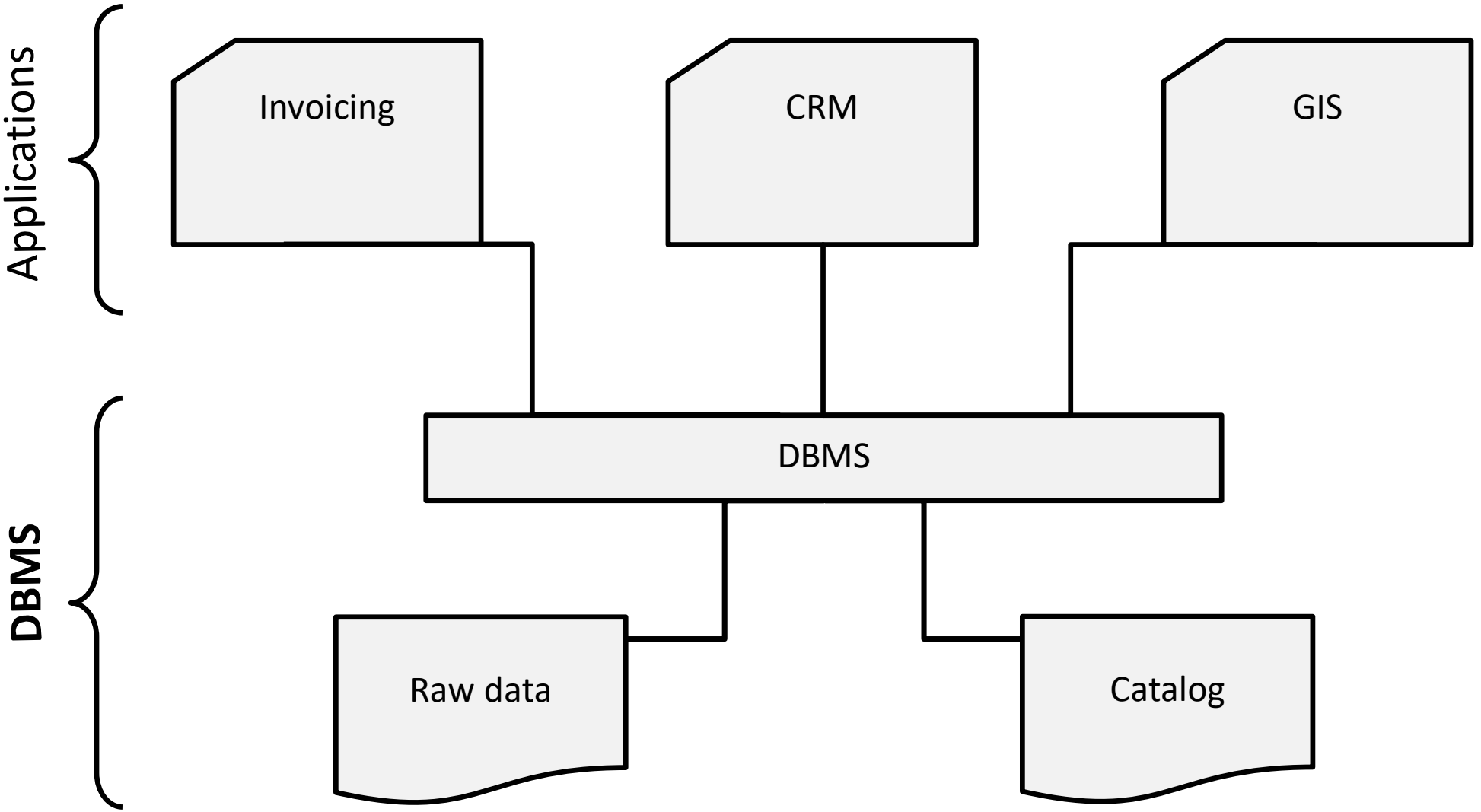


Podejście oparte o pliki vs bazodanowe

Początki: aplikacje bazodanowe budowane bezpośrednio na *systemach plików*, co prowadziło do:

- Przechowywanie zduplikowanych lub nadmiarowych informacji
- Niebezpieczeństwo niespójnych danych
- Silne powiązanie między danymi i aplikacjami
- Trudno zarządzać kontrolą współbieżności
- Aplikacje trudne do zintegrowania

Podejście oparte o pliki vs bazodanowe



Podejście oparte o pliki vs bazodanowe

- Podejście oparte o bazę danych
  - przewyższa podejście oparte o pliki pod względem wydajności, spójności i zarządzania
  - luźne połączenie między aplikacjami a danymi
  - zapewnione udogodnienia do odpytywania i pobierania danych

Podejście oparte o pliki vs bazodanowe

- O pliki

```
Procedure FindCustomer;
begin
     open file Customer.txt;
Read(Customer)
While not EOF(Customer)
If Customer.name='Bart' then
display(Customer);
EndIf
Read(Customer);
EndWhile;
End;
```

- O bazę danych (SQL)

```
SELECT *
FROM Customer
WHERE
name = 'Bart'
```

Elementy systemu baz danych

- Model bazy danych a instancje
- Modele danych
- Architektura trójwarstwowa
- Rola katalogu
- Użytkownicy bazy danych
- Języki bazodanowe

Model bazy danych a instancje

- Model bazy danych lub schemat bazy danych dostarcza opisu danych w bazie danych na różnych poziomach szczegółowości i specyfikuje różne elementy danych, ich cechy i zależności, ograniczenia, szczegóły dotyczące przechowywania, itp
  - określony podczas projektowania bazy danych i nie oczekuje się, że będzie się zbyt często zmieniać
  - przechowywany w katalogu
- Stan bazy danych reprezentuje dane w bazie danych w określonym momencie
  - zwany także bieżącym zbiorem instancji
  - zazwyczaj zmienia się na bieżąco

Model bazy danych a instancje

- Model bazy danych

  Student (number, name, address, email)

  Course (number, name)

  Building (number, address)

Model bazy danych a instancje

- Stan bazy danych

STUDENT			
Number	**Name**	**Address**	**Email**
0165854	Bart Baesens	1040 Market Street, SF	Bart.Baesens@kuleuven.be
0168975	Seppe vanden Broucke	520, Fifth Avenue, NY	Seppe.vandenbroucke@kuleuven.be
0157895	Wilfried Lemahieu	644, Wacker Drive, Chicago	Wilfried.Lemahieu@kuleuven.be

COURSE	
Number	**Name**
D0I69A	Principles of Database Management
D0R04A	Basic Programming
D0T21A	Big Data & Analytics

BUILDING	
Number	**Address**
0600	Naamsestraat 69, Leuven
0365	Naamsestraat 78, Leuven
0589	Tiensestraat 115, Leuven

Model danych

- Model bazy danych składa się z różnych modeli danych, z których każdy opisuje dane z różnych perspektyw
- Model danych zapewnia jasny i jednoznaczny opis elementów danych, ich relacji i różnych ograniczeń danych z określonej perspektywy

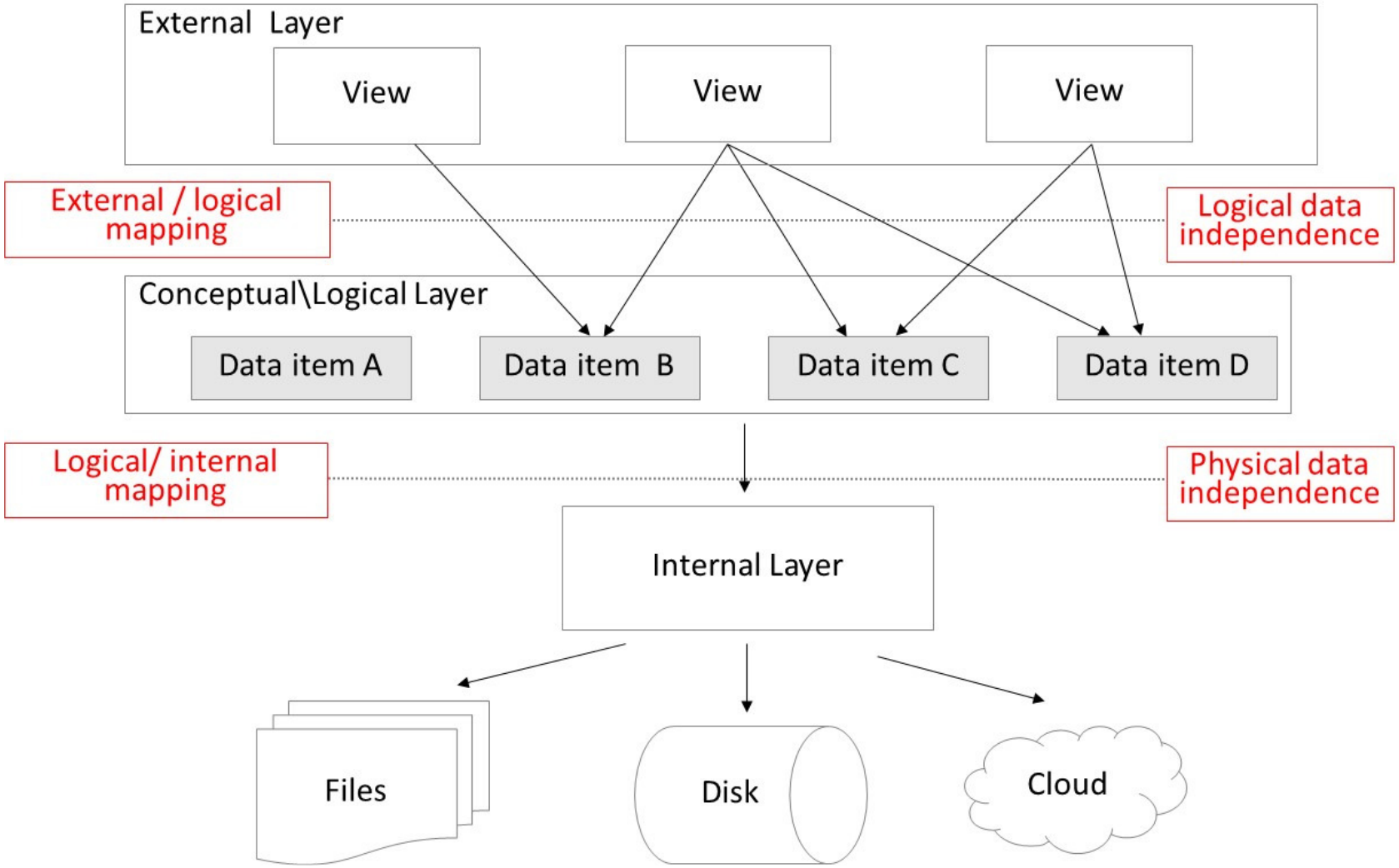
Model danych

- *Koncepcyjny model danych* zapewnia wysokopoziomowy opis elementów danych wraz z ich charakterystykami i relacjami
  - instrument komunikacji pomiędzy architektem informacji a użytkownikiem biznesowym
  - powinien być niezależny od wdrożenia, przyjazny dla użytkownika i zbliżony do tego, jak użytkownik biznesowy postrzega dane
  - Zwykle reprezentowany przez model ER (EER) lub modelu zorientowanego obiektowo (UML)
- *Logiczny model danych* to translacja lub mapowanie konceptualnego modelu danych na określone środowisko implementacyjne
  - model hierarchiczny, CODASYL, relacyjny, obiektowy, rozszerzony relacyjny, XML lub NoSQL

Model danych

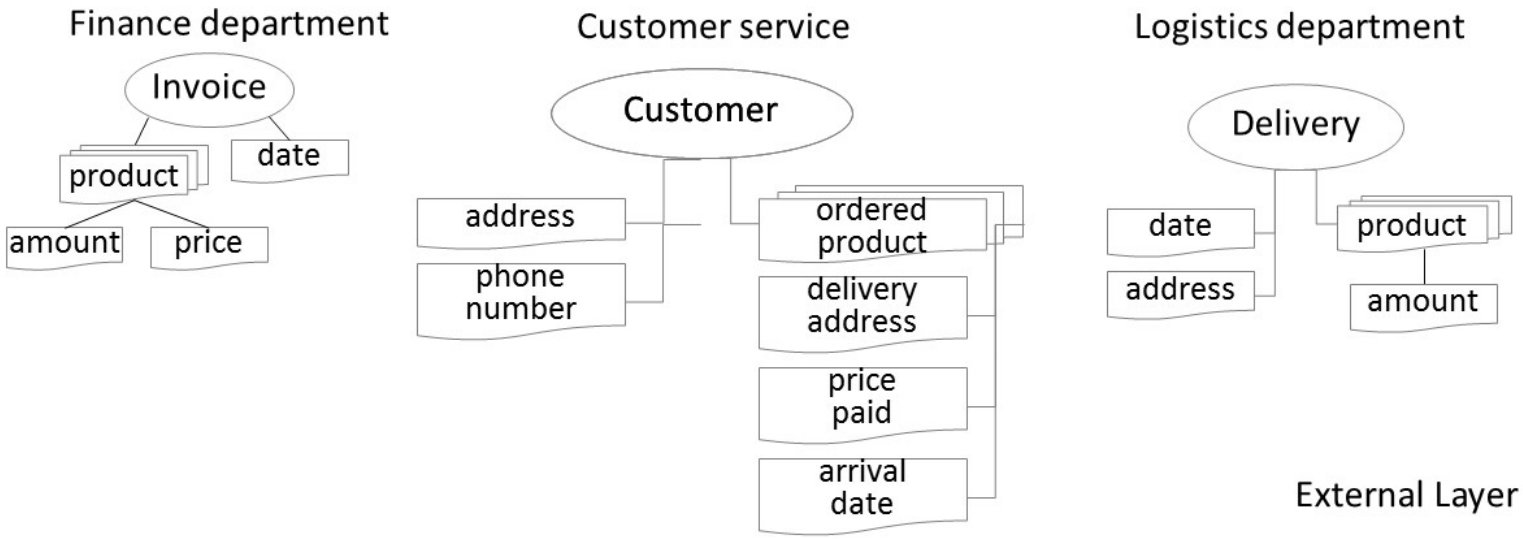
- *Logiczny model danych* można zmapować do wewnętrznego modelu danych, który reprezentuje fizyczne szczegóły przechowywania danych
  - jasno opisuje, które dane są przechowywane, gdzie, w jakim formacie, jakie indeksy są dostarczane w celu przyspieszenia wyszukiwania itp.
  - ściśle związany z DBMS
- *Zewnętrzny model danych* zawiera różne podzbiory elementów danych w modelu logicznym, zwane również widokami, dostosowane do potrzeb konkretnych aplikacji lub grup użytkowników

Architektura trójwarstwowa

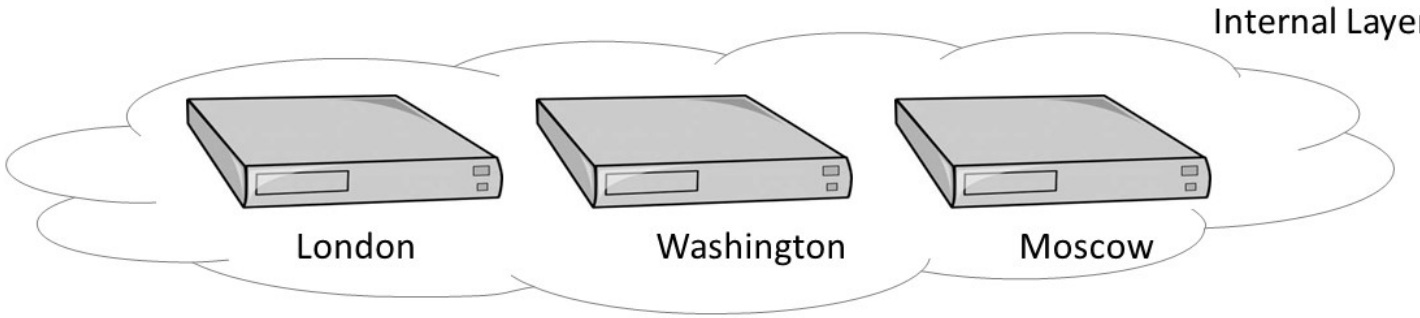


dane fizyczne + logiczna niezależność danych

Architektura trójwarstwowa



Product	name, description, cost, …	Conceptual\Logical Layer
Customer	name, phone, address, ...	
Invoice	customer, date, products (with price and amount), …	
Delivery	invoice, address, date, ...	



Katalog

- Jądro DBMS
- Zawiera definicje danych oraz metadane
- Przechowuje definicje widoków, logiczne i wewnętrzne modele danych oraz synchronizuje te trzy modele danych, aby zapewnić ich spójność

Użytkownicy bazy danych

- Architekt informacji projektuje koncepcyjny model danych
  - ściśle współdziała z użytkownikiem biznesowym
- Projektant bazy danych przekłada koncepcyjny model danych na logiczny i wewnętrzny model danych
- Administrator bazy danych (DBA) odpowiada za wdrożenie i monitorowanie bazy danych
- Twórca aplikacji tworzy aplikacje bazodanowe w języku programowania takim jak Java czy Python
- Użytkownik biznesowy uruchamia aplikacje, aby wykonać określone operacje na bazie danych

Języki bazodanowe

- Data Definition Language (DDL) jest używany przez DBA do wyrażenia zewnętrznych, logicznych i wewnętrznych modeli danych bazy danych
  - definicje są przechowywane w katalogu
- Data Manipulation Language (DML) służy do pobierania, wstawiania, usuwania i modyfikowania danych
  - Instrukcje DML mogą być osadzone w języku programowania lub wprowadzane interaktywnie za pomocą narzędzia front-end do wysyłania zapytań
- Structured Query Language (SQL) oferuje zarówno instrukcje DDL, jak i DML dla relacyjnych systemów baz danych

Zalety systemów bazodanowych i zarządzania bazami danych

- Niezależność danych
- Modelowanie baz danych
- Zarządzanie danymi ustrukturyzowanymi, częściowo ustrukturyzowanymi i nieustrukturyzowanymi
- Zarządzanie nadmiarowością danych
- Określanie zasad integralności
- Kontrola współbieżności
- Funkcje tworzenia kopii zapasowych i odzyskiwania
- Bezpieczeństwo danych
- Narzędzia wydajności

Niezależność danych

- Niezależność danych oznacza, że zmiany w definicjach danych mają minimalny lub żaden wpływ na aplikacje
- *Fizyczna niezależność* danych oznacza, że ani aplikacje, ani widoki ani logiczny model danych nie muszą być zmieniane, gdy wprowadzane są zmiany w specyfikacjach przechowywania danych w wewnętrznym modelu danych
  - DBMS powinien zapewniać interfejsy między logicznymi a wewnętrznymi modelami danych
- *Logiczna niezależność danych* oznacza, że na aplikacje w minimalnym stopniu wpływają zmiany w koncepcyjnym lub logicznym modelu danych
  - widoki w zewnętrznym modelu danych będą działać jak tarcza ochronna
  - DBMS musi zapewniać interfejsy między warstwą koncepcyjną/logiczną a zewnętrzną

Modelowanie baz danych

- Model danych to jawna reprezentacja elementów danych wraz z ich charakterystykami i zależnościami
- Koncepcyjny model danych powinien zapewniać formalne i doskonałe odwzorowanie wymagań dotyczących danych w procesie biznesowym i jest tworzony we współpracy z użytkownikiem biznesowym
  - przekładany na logiczny i wewnętrzny model danych
- Ważne, aby założenia i niedociągnięcia modelu danych były jasno udokumentowane

Zarządzanie danymi ustrukturyzowanymi, częściowo ustrukturyzowanymi i nieustrukturyzowanymi

- Dane ustrukturyzowane
  - można opisać zgodnie z formalnym logicznym modelem danych
  - możliwość wyrażania reguł integralności i egzekwowania poprawności danych
  - ułatwia również wyszukiwanie, przetwarzanie i analizę danych
  - np. identyfikator, nazwisko i adres studenta
- Dane nieustrukturyzowane
  - brak drobnoziarnistych komponentów w pliku lub serii znaków, które mogą być interpretowane w znaczący sposób przez DBMS lub aplikację
  - np. dokument z biografiami znanych obywateli Krakowa
  - Uwaga: ilość danych nieustrukturyzowanych przewyższa ilość danych ustrukturyzowanych

Zarządzanie danymi ustrukturyzowanymi, częściowo ustrukturyzowanymi i nieustrukturyzowanymi

- Dane częściowo ustrukturyzowane
  - dane, które mają określoną strukturę, ale struktura może być bardzo nieregularna lub bardzo niestabilna
  - Np. strony internetowe poszczególnych użytkowników na platformie mediów społecznościowych lub dokumenty życiorysów w bazie danych HR

Zarządzanie nadmiarowością danych

- Powielanie danych może być pożądane w środowiskach rozproszonych w celu poprawy wydajności odzyskiwania danych
- DBMS jest teraz odpowiedzialny za zarządzanie redundancją poprzez zapewnienie możliwości synchronizacji w celu zapewnienia spójności danych
- W porównaniu z podejściem plikowym, DBMS gwarantuje poprawność danych bez ingerencji użytkownika

Określanie reguł integralności

- Reguły syntaktyczne określają, w jaki sposób dane powinny być reprezentowane i przechowywane
  - Np. IDKlienta jest liczbą całkowitą; data urodzenia powinna być przechowywana w formacie miesiąc, dzień i rok
- Reguły semantyczne skupiają się na poprawności semantycznej lub znaczeniu danych
  - Np. identyfikator klienta jest unikalny; saldo konta powinno być > 0; klient nie może zostać usunięty, jeśli ma oczekujące faktury
- Reguły integralności są określone jako część koncepcyjnego/logicznego modelu danych i przechowywane w katalogu
  - bezpośrednio wymuszane przez DBMS zamiast aplikacji

Kontrola współbieżności

- DBMS ma wbudowane udogodnienia do obsługi współbieżnego lub równoległego wykonywania programów bazodanowych
- Kluczowym pojęciem jest transakcja bazy danych
  - sekwencja operacji odczytu/zapisu uważana za jednostkę atomową w tym sensie, że albo wszystkie operacje są wykonywane, albo wcale
- Operacje odczytu/zapisu mogą być wykonywane w tym samym czasie przez DBMS
- DBMS powinien unikać niespójności

Kontrola współbieżności

- Problem utraconych aktualizacji

Time	**T1**	**T2**	**balance**
t1		Begin transaction	$100
t2	Begin transaction	read(balance)	$100
t3	read(balance)	balance=balance+120	$100
t4	balance=balance-50	write(balance)	$220
t5	write(balance)	End transaction	$50
t6	End transaction		$50

Kontrola współbieżności

- DBMS musi wspierać własność ACID (Atomicity, Consistency, Isolation, Durability)
  - Atomowość wymaga, aby transakcja została wykonana w całości lub wcale
  - Spójność zapewnia, że transakcja przenosi bazę danych z jednego spójnego stanu do drugiego
  - Izolacja zapewnia, że efekt równoczesnych transakcji powinien być taki sam, jak gdyby były wykonywane w izolacji
  - Trwałość zapewnia, że zmiany w bazie danych dokonane przez transakcję uznaną za udaną muszą być trwałe w każdych okolicznościach

Funkcje tworzenia kopii zapasowych i odzyskiwania

- Funkcje tworzenia kopii zapasowych i odzyskiwania mogą być wykorzystywane do radzenia sobie ze skutkami utraty danych spowodowanych błędami sprzętowymi lub sieciowymi lub błędami w systemie lub oprogramowaniu
- Można wykonywać pełną lub przyrostową kopię zapasową
- Możliwości odtwarzania pozwalają na przywrócenie danych do stanu poprzedniego po ich utracie lub uszkodzeniu

Bezpieczeństwo danych

- Bezpieczeństwo danych może być egzekwowane przez DBMS
- Niektórzy użytkownicy mają dostęp do odczytu, podczas gdy inni mają dostęp do zapisu danych (funkcja oparta na rolach)
  - Np. zapasy zarządzane przez dostawcę (VMI)
- Dostęp do danych może być zarządzany za pomocą loginów i haseł przypisanych do użytkowników lub kont użytkowników
- Każde konto ma własne reguły autoryzacji, które można przechowywać w katalogu

Narzędzia wydajności

- Trzy KPI dla DBMS to
  - czas odpowiedzi oznaczający czas, jaki upłynął od wysłania żądania do bazy danych do jego pomyślnego zakończenia
  - przepustowość reprezentująca transakcje, które DBMS może przetwarzać w jednostce czasu
  - wykorzystanie przestrzeni w odniesieniu do przestrzeni wykorzystywanej przez DBMS do przechowywania zarówno surowych danych, jak i metadanych
- DBMS są dostarczane z różnymi rodzajami narzędzi mających na celu poprawę tych KPI
  - Np. narzędzia do dystrybucji i optymalizacji przechowywania danych, dostrajania indeksów w celu szybszego wykonywania zapytań, dostrajania zapytań w celu poprawy wydajności aplikacji lub optymalizacji zarządzania buforami