

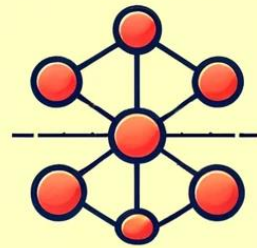
Temat 10.

Bazy danych, hurtownie
danych i Big Date

dr inż. Michał Malinowski

<https://www.drmalinowski.edu.pl>

WPROWADZENIE
DO INFORMATYKI



DEFINE NODES
REFATTIOWIPS



MODEL C
AREFITEICSTOHP S



MOLEL NODOS
AND EI OCTIHIRE



APPLYING INTIDEES
Q KUNRONUNTS



APPLIVIG INDEXDES
& ORSSTUCINS



IMPOR IG
CORARCAT



SUIDATING THE GRAP I



oTSGAD

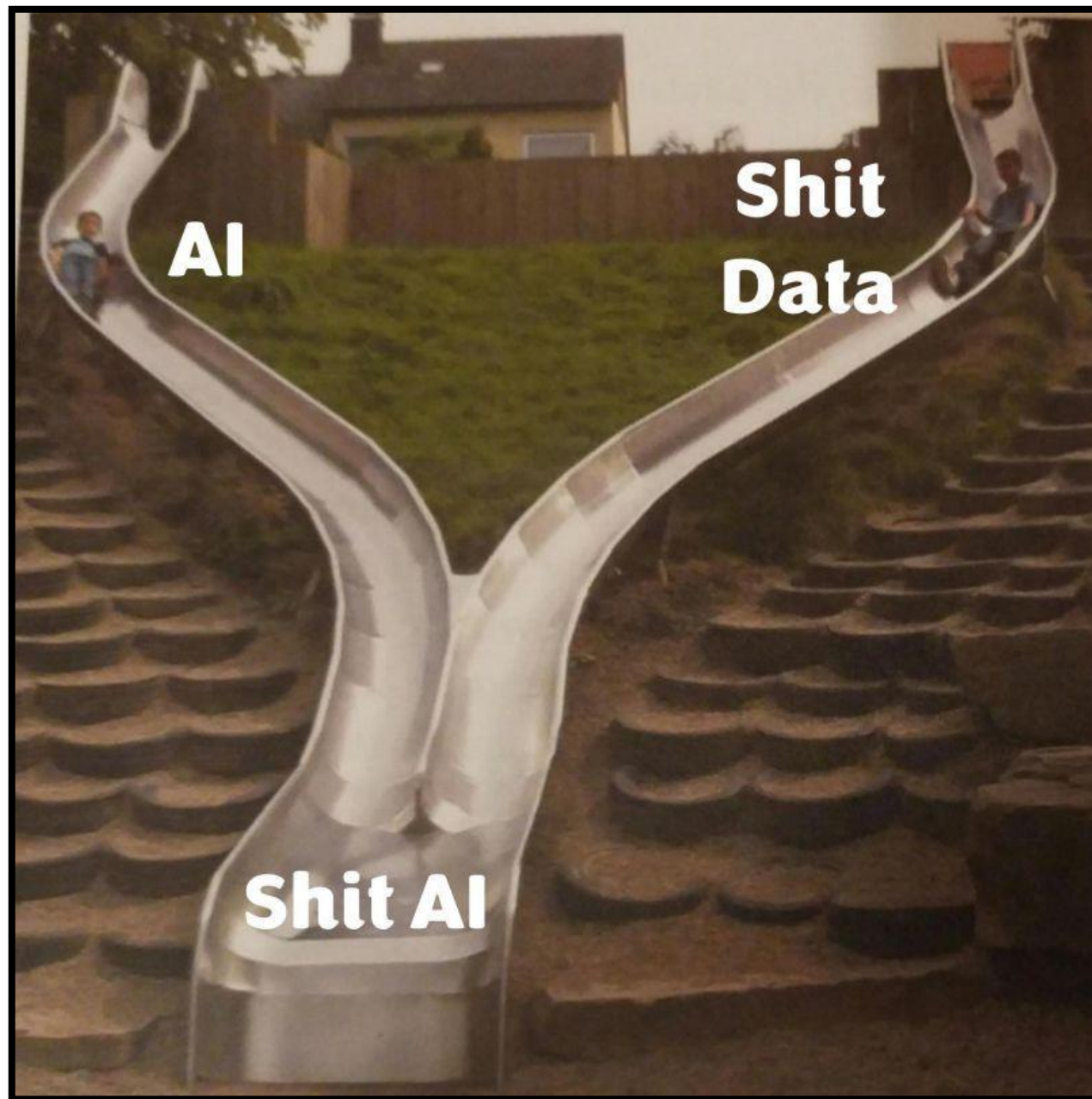


VALIDATINOE GRAPH

- Definicja i znaczenie baz danych
- Historia rozwoju baz danych
- Modele baz danych
- Bazy klucz-wartość
- Bazy grafowe
- Model relacyjny
- Hurtownie danych
- Big Date



Bazy danych to pokarm dla sztucznej inteligencji.



Definicja i znaczenie baz danych

Baza danych

to zorganizowany zbiór danych, który umożliwia łatwe przechowywanie, zarządzanie, dostęp do informacji oraz dane są zorganizowane w sposób strukturalny.

Biblioteki

- Katalogi kartkowe
- Systemy klasyfikacji

Rejestry publiczne

- Księgi wieczyste
- Akta urodzeń, małżeństw i zgonów

Kroniki i Archiwa

- Kroniki klasztorne
- Archiwa państwowe

Zbiory danych handlowych

- Księgi rachunkowe
- Rejestry magazynowe

Zbiory danych naukowych

- Zielniki Botaniczne
- Tablice astronomiczne

Indeksy w archiwach kościelnych

- Rejestry parafialne

Spisy ludności

- Manuskrypty spisów
- Dane statystyczne

Cyfrowe

- Organizacji i Państwowe
- Prywatne

Definicja i znaczenie baz danych

Rola baz danych w nowoczesnych systemach informatycznych

Centralne repozytorium danych:

- Przechowuje duże ilości danych w jednym miejscu, umożliwiając łatwy dostęp i zarządzanie.

Wspomaganie podejmowania decyzji:

- Bazy danych są kluczowe dla analiz biznesowych, raportowania oraz podejmowania decyzji opartych na danych.

Zwiększenie efektywności operacyjnej:

- Automatyzacja procesów biznesowych i operacji dzięki sprawnemu zarządzaniu danymi.

Ułatwienie integracji systemów:

- Bazy danych umożliwiają integrację różnych aplikacji i systemów poprzez wspólny dostęp do danych.

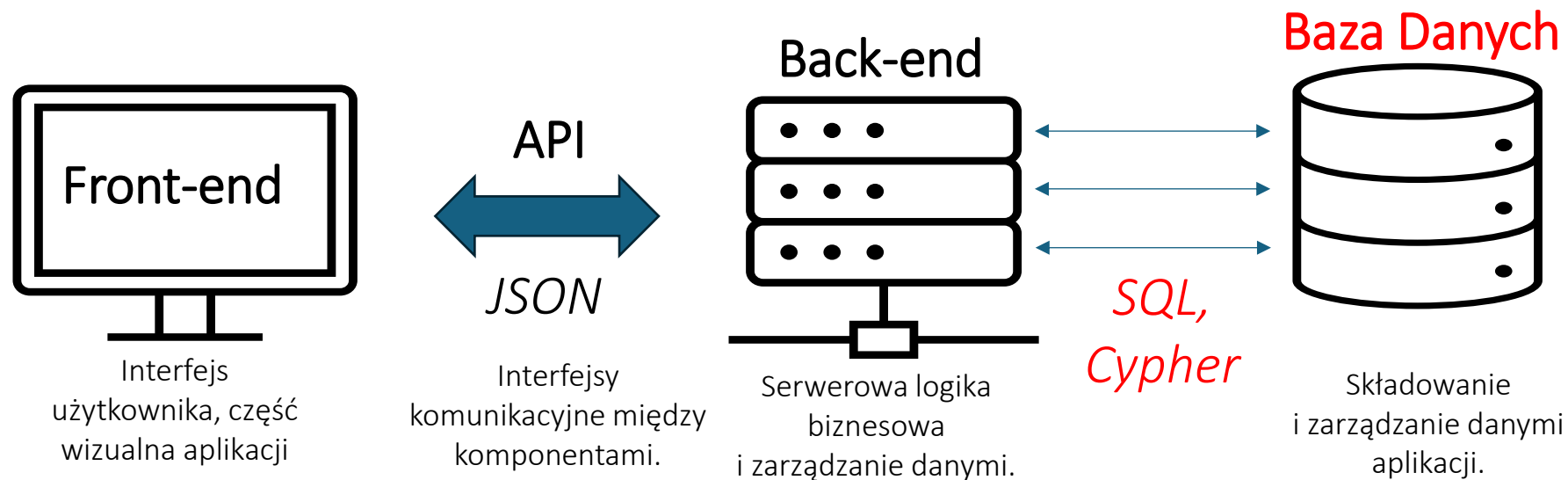
Bezpieczeństwo i ochrona danych:

- Bazy danych oferują mechanizmy zabezpieczające, takie jak kontrola dostępu, szyfrowanie oraz audyt.

Definicja i znaczenie baz danych

Miejsce baz danych w architekturze aplikacji

Składowanie i zarządzanie danymi aplikacji

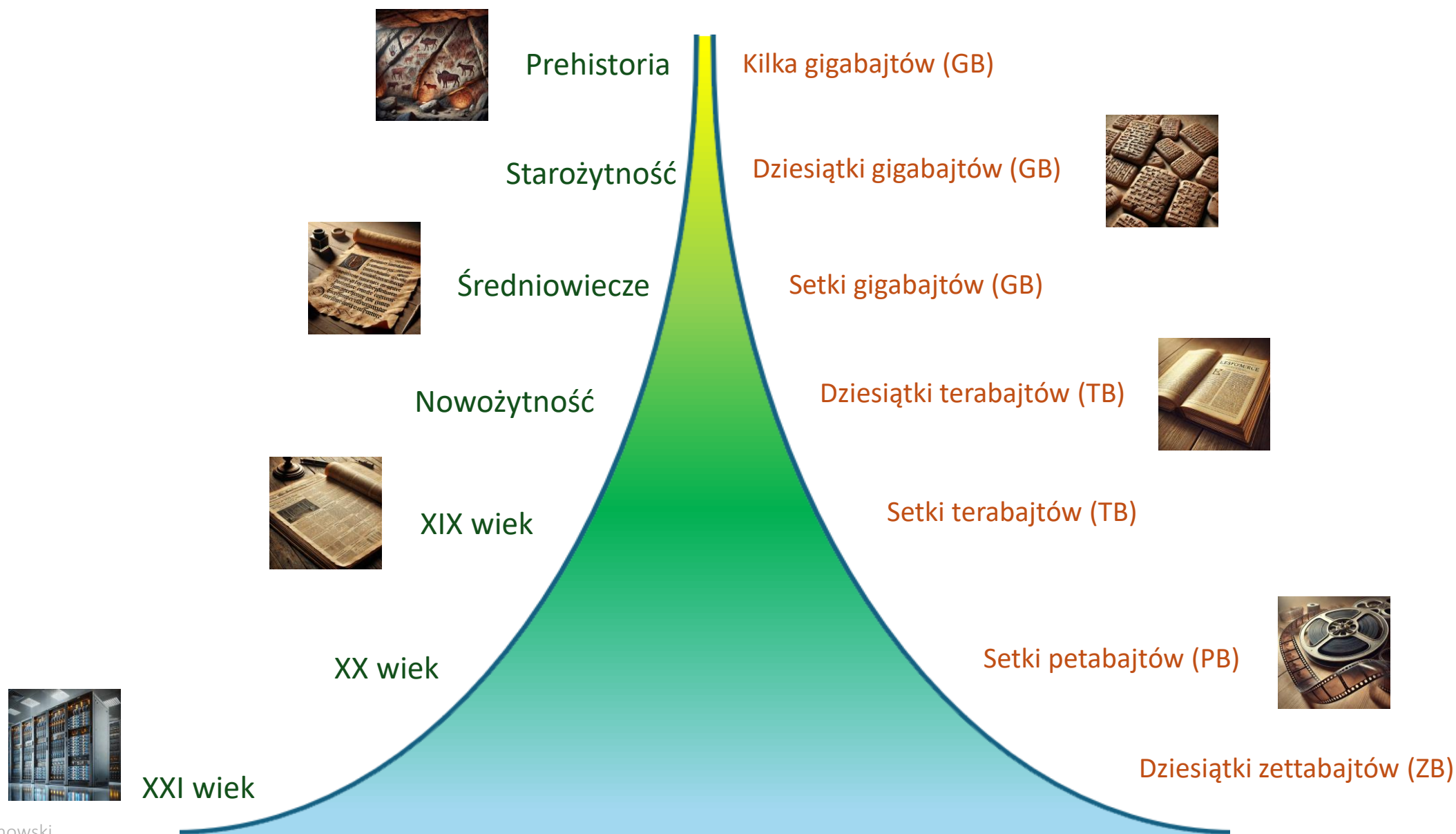


Historia rozwoju baz danych

Nośniki i pozyskanie informacji na przestrzeni historii

Okres	Nośniki informacji	Metoda pozyskania informacji	Szacunkowa wielkość zachowanych danych
Prehistoria	Pamięć plemienna, ściany jaskiń	Pytania do rodziny lub wodza plemienia	Kilka gigabajtów (GB)
Starożytność	Gliniane tabliczki, papirus	Konsultacje z uczonymi lub kapłanami	Dziesiątki gigabajtów (GB)
Średniowiecze	Pergamin, skóra	Manualne przeglądanie skryptoriów w klasztorach	Setki gigabajtów (GB)
Nowożytność	Książki (papier)	Korzystanie z drukowanych katalogów bibliotecznych	Dziesiątki terabajtów (TB)
XIX wiek	Gazety (papier)	Kluby kawiarnie jako miejsca spotkań i dyskusji intelektualistów	Setki terabajtów (TB)
XX wiek	Film i nagrania, Nośniki cyfrowe, Strony WWW	Telewizja i radio. Wyszukiwanie za pomocą internetowych wyszukiwarek	Setki petabajtów (PB)
XXI wiek	SocialMedia, BigData, Chmura	Wyszukiwanie za pomocą sztucznej inteligencji	Dziesiątki zettabajtów (ZB)

Ilość i nośniki danych na przestrzeni historii



Historia rozwoju baz danych

Historia



Systemy Plikowe

- Dane przechowywane w osobnych plikach na taśmach i dyskach.
- Problemy z duplikacją, integralnością, trudności w zarządzaniu.

1950-1960



Hierarchiczne i Sieciowe Bazy Danych

- Model hierarchiczny: Dane zorganizowane w strukturę drzewa (np. IMS od IBM).
- Model sieciowy: Złożone sieci zależności między danymi (np. CODASYL DBTG).
- Skomplikowane struktury, trudne do modyfikacji.

1960-1970



Relacyjne Bazy Danych

- Teoria relacyjna: Wprowadzona przez Edgara F. Codd, oparta na algebrze relacyjnej.
- Model relacyjny: Dane przechowywane w tabelach, klucze jako relacje między danymi.
- Pojawienie się RDBMS (np. Oracle, IBM DB2, MS SQL Server).

1970-1980



Nowoczesne Relacyjne Bazy Danych

- Wsparcie dla transakcji, integralności danych, zaawansowane indeksowanie.
- Rozwój standardu SQL (procedury składowane, wyzwalacze, widoki).
- Wprowadzenie OLAP, hurtowni danych, migracja do chmury (np. Amazon RDS).

1980-obecnie



Alternatywne Modele Danych i NoSQL

- NoSQL: Elastyczne rozwiązania dla dużych zbiorów danych (np. MongoDB, Cassandra).
- Bazy grafowe: Przetwarzanie złożonych relacji (np. Neo4j).
- Przewidywany rozwój w kierunku hybrydowych modeli i integracji z AI.

2000-obecnie

Historia rozwoju baz danych

Współczesne Bazy Danych

Typ Bazy Danych	Nazwa	Opis
Relacyjne	MySQL	Open-source system zarządzania bazą danych, popularny w aplikacjach webowych.
	Oracle Database	Jeden z najbardziej zaawansowanych systemów baz danych, używany w dużych aplikacjach korporacyjnych.
	Microsoft SQL Server	System zarządzania bazą danych opracowany przez Microsoft, szeroko stosowany w korporacjach.
	PostgreSQL	Zaawansowany system zarządzania bazą danych, znany z obsługi skomplikowanych zapytań SQL.
	SQLite	Lekka, wbudowana baza danych, idealna do aplikacji mobilnych i desktopowych.
Nierelacyjne (NoSQL)	Neo4j	Baza danych grafowa, która przechowuje dane w postaci węzłów, relacji i właściwości.
	MongoDB	Baza danych dokumentowa, która przechowuje dane w formacie JSON.
	Cassandra	Baza danych kolumnowa, zaprojektowana do obsługi dużych ilości danych rozproszonych.
	Redis	Baza danych klucz-wartość, znana z wysokiej wydajności i wsparcia dla różnych struktur danych.
	CouchDB	Baza danych dokumentowa, która przechowuje dane w formacie JSON z obsługą wersjonowania.

Ranking - Systemy baz danych

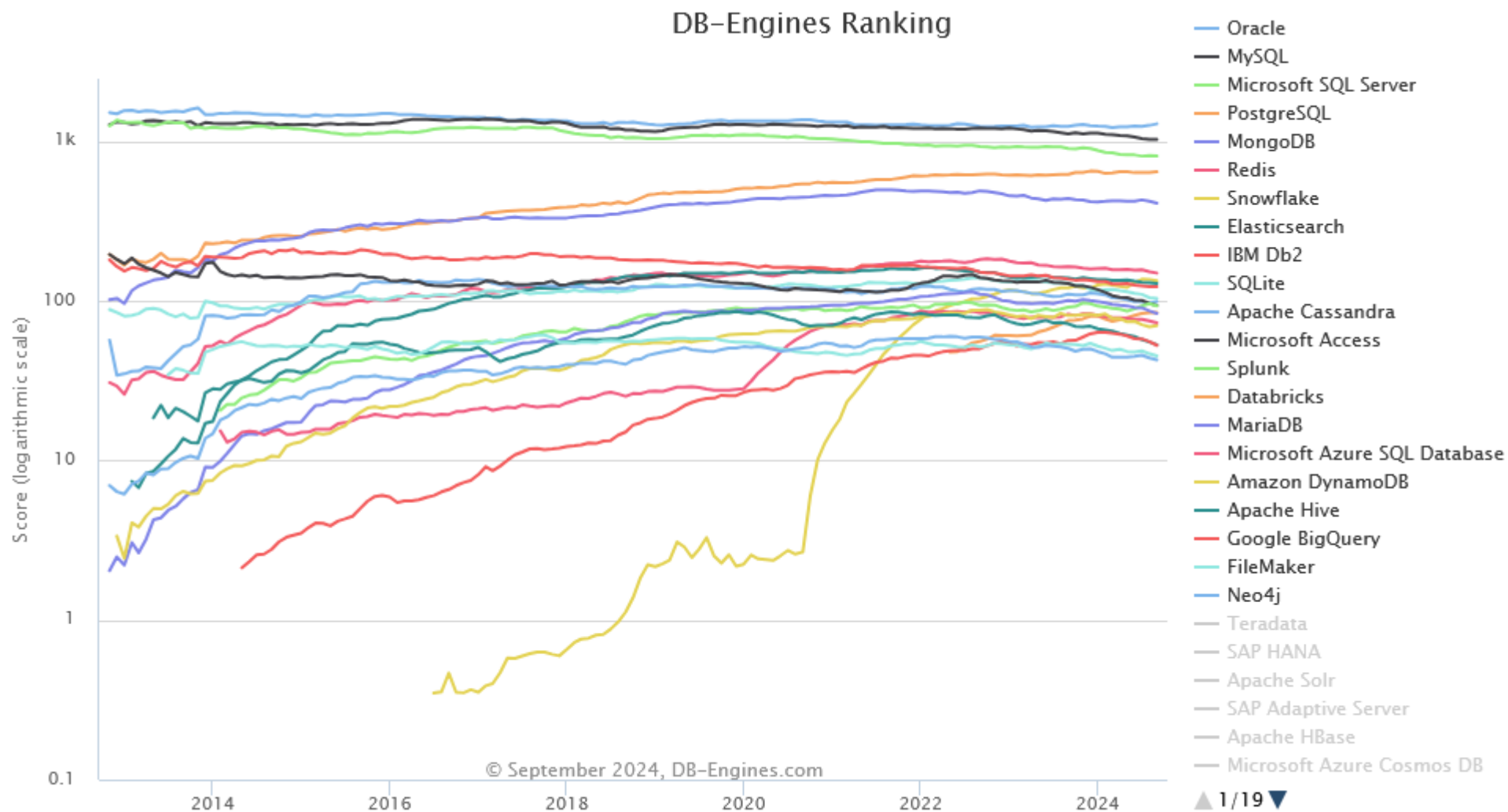
423 systems in ranking, September 2024

Rank			DBMS	Database Model	Score		
Sep 2024	Aug 2024	Sep 2023			Sep 2024	Aug 2024	Sep 2023
1.	1.	1.	Oracle	Relational, Multi-model	1286.59	+28.11	+45.72
2.	2.	2.	MySQL	Relational, Multi-model	1029.49	+2.63	-82.00
3.	3.	3.	Microsoft SQL Server	Relational, Multi-model	807.76	-7.41	-94.45
4.	4.	4.	PostgreSQL	Relational, Multi-model	644.36	+6.97	+23.61
5.	5.	5.	MongoDB	Document, Multi-model	410.24	-10.74	-29.18
6.	6.	6.	Redis	Key-value, Multi-model	149.43	-3.28	-14.26
7.	7.	11.	Snowflake	Relational	133.72	-2.25	+12.83
8.	8.	7.	Elasticsearch	Search engine, Multi-model	128.79	-1.04	-10.20
9.	9.	8.	IBM Db2	Relational, Multi-model	123.05	+0.04	-13.67
10.	10.	9.	SQLite	Relational	103.35	-1.44	-25.85
11.	11.	12.	Apache Cassandra	Wide column, Multi-model	98.94	+1.94	-11.11
12.	12.	10.	Microsoft Access	Relational	93.76	-2.61	-34.81
13.	13.	14.	Splunk	Search engine	93.02	-3.08	+1.63
14.	15.	17.	Databricks	Multi-model	84.24	-0.22	+9.06
15.	14.	13.	MariaDB	Relational, Multi-model	83.44	-3.09	-17.01
16.	16.	15.	Microsoft Azure SQL Database	Relational, Multi-model	72.95	-2.08	-9.78
17.	17.	16.	Amazon DynamoDB	Multi-model	70.06	+1.15	-10.85
18.	19.	18.	Apache Hive	Relational	53.07	-2.17	-18.76
19.	18.	20.	Google BigQuery	Relational	52.67	-2.86	-3.80
20.	20.	21.	FileMaker	Relational	45.20	-1.47	-8.40
21.	21.	23.	Neo4j	Graph	42.68	-1.22	-7.71
22.	23.	19.	Teradata	Relational, Multi-model	41.47	-0.78	-18.86
23.	22.	22.	SAP HANA	Relational, Multi-model	41.32	-1.04	-9.29
24.	24.	24.	Apache Solr	Search engine, Multi-model	34.15	-2.14	-12.94
25.	25.	25.	SAP Adaptive Server	Relational, Multi-model	32.67	-0.61	-10.64

DB-Engines Ranking

<https://db-engines.com/en/ranking>

Ranking - Systemy baz danych



Modele baz danych

SQL (80%)

Bazy relacyjne *

Przechowują dane w tabelach powiązanych kluczami głównymi i obcymi, zapewniając integralność i spójność danych.

Przykłady: Oracle, MSSQL, MySQL

Zastosowanie: Systemy transakcyjne, raportowanie, analizy danych

NoSQL (20%)

Bazy dokumentowe

Przechowują dane w formie dokumentów, zazwyczaj w formacie JSON, BSON lub XML.

Przykłady: MongoDB, CouchDB.

Zastosowanie: Idealne do przechowywania danych o zmiennej strukturze, takich jak dokumenty, profile użytkowników.

Bazy klucz-wartość

Przechowują dane jako pary klucz-wartość, gdzie klucz jest unikalnym identyfikatorem, a wartość może być dowolnym typem danych.

Przykłady: Redis, DynamoDB.

Zastosowanie: Świetne do szybkiego przechowywania i pobierania dużych ilości danych, takich jak sesje użytkowników, dane cache.

Bazy kolumnowe

Organizują dane w kolumnach zamiast w wierszach, co umożliwia efektywne przechowywanie i przetwarzanie dużych zbiorów danych analitycznych.

Przykłady: Apache Cassandra, HBase.

Zastosowanie: Idealne do analizy dużych zestawów danych, gdzie istotne są operacje na kolumnach danych.

Bazy grafowe *

Przechowują dane w postaci węzłów, krawędzi i właściwości, co umożliwia modelowanie złożonych relacji między danymi.

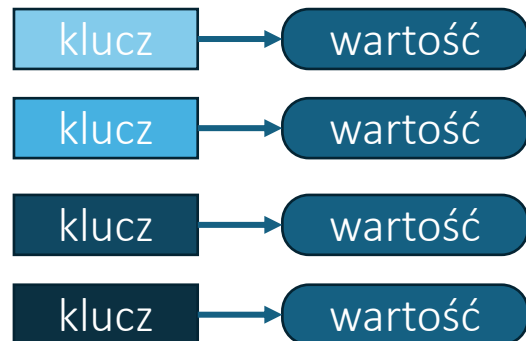
Przykłady: Neo4j, ArangoDB.

Zastosowanie: Używane w aplikacjach wymagających analizy grafów, takich jak sieci społecznościowe, zarządzanie zasobami, rekomendacje.

Bazy NoSQL

Bazy klucz-wartość
(Key-Value): gdzie wszystko przechowywane jest jako para klucz-wartość.
Przykłady: Redis, Amazon DynamoDB

Bazy klucz-wartość

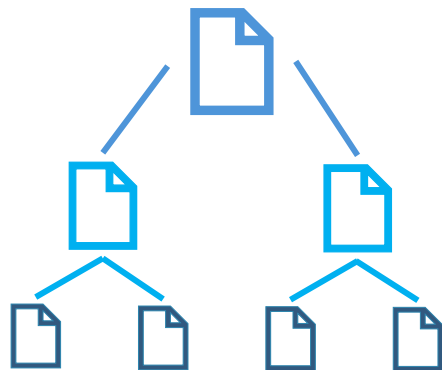


Bazy rodziny kolumn

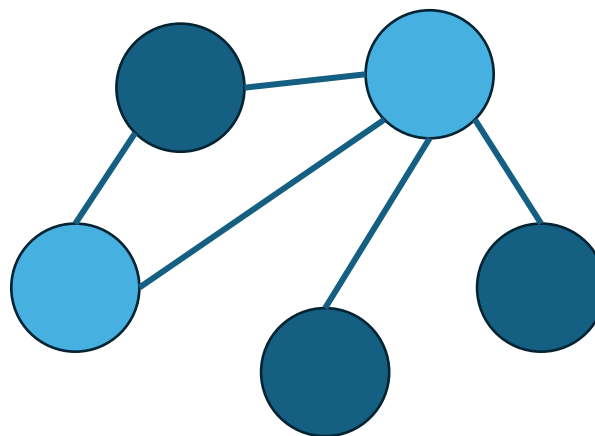
Bazy kolumnowe
(Columnar): dane są przechowywane i wyszukiwane na podstawie kolumn, a nie wierszy.
Przykłady: Apache Cassandra, Google Bigtable.

Bazy dokumentowe
(Document): przechowują informacje jako dokumenty, często w formatach takich jak JSON.
Przykłady: MongoDB, Couchbase.

Bazy dokumentów



Bazy grafowe



Bazy grafowe
(Graph): skoncentrowane na przechowywaniu informacji o węzłach i relacjach między nimi.
Przykład: Neo4j, JanusGraph

Bazy klucz-wartość

Bazy klucz-wartość

to jeden z najprostszych i najbardziej efektywnych modeli przechowywania danych, który polega na skojarzeniu unikalnego klucza z odpowiadającą mu wartością.

Chociaż ich struktura jest wyjątkowo prosta, bazy te stanowią fundament dla wielu zaawansowanych systemów, które obsługują ogromne ilości danych w czasie rzeczywistym.

Przestrzeń: Klient		Przestrzeń: Piłkarz		Przestrzeń: Polityk	
osoba:1	Nowak	osoba:1	Smolarek	osoba:1	Sikorski
osoba:2	Kowalski	osoba:2	Boniek	osoba:2	Kuroń
osoba:3	Malinowski	osoba:3	Lato	osoba:3	Mazowiecki



Bazy klucz-wartość

Zalety i wady

Szybkość działania:

- Bazy klucz-wartość oferują bardzo szybkie operacje odczytu i zapisu, ponieważ dane są bezpośrednio dostępne przez klucz.

Prostota i elastyczność:

- Brak schematu pozwala na łatwe przechowywanie danych o dowolnej strukturze, co jest szczególnie przydatne w dynamicznych aplikacjach.

Łatwe skalowanie poziome:

- Rozproszone bazy klucz-wartość (np. Redis, DynamoDB) można łatwo skalować, co pozwala obsłużyć ogromne ilości danych i ruchu.

Lekkość systemu:

- Ze względu na prostą strukturę, bazy te mają niskie wymagania zasobów i są łatwe w implementacji w różnych środowiskach.

Brak zaawansowanych operacji:

- Nie obsługują zapytań typu SQL ani relacji między danymi, co ogranicza ich zastosowanie w bardziej skomplikowanych scenariuszach.

Trudności w zarządzaniu złożonymi danymi:

- Wartości przechowywane w bazie mogą być trudne do analizy, jeśli są złożonymi strukturami, np. zagnieżdżonymi obiektami.

Ograniczone funkcje indeksowania:

- Brak możliwości tworzenia indeksów na danych innych niż klucze, co może spowolnić wyszukiwanie w dużych zbiorach.

Potencjalna redundancja danych:

- W przypadku przechowywania powiązanych informacji konieczne jest duplikowanie danych, ponieważ brakuje mechanizmu relacji, co może prowadzić do problemów z synchronizacją.

Bazy grafowe

Grafowa baza danych

to system zarządzania bazą danych zaprojektowany do przechowywania, zarządzania i wyszukiwania informacji w strukturach grafowych.



Elastyczność

Łatwość modelowania
złożonych relacji
i dynamicznych
schematów.



Wydajność

Szybka nawigacja
i zapytania w
złożonych sieciach
relacji.



Intuicyjność

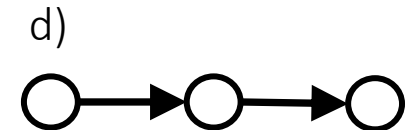
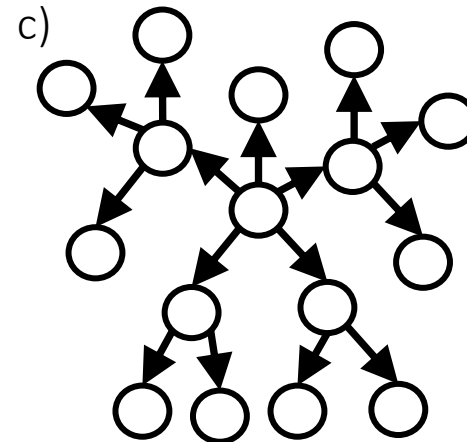
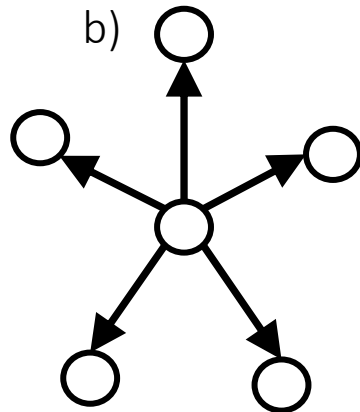
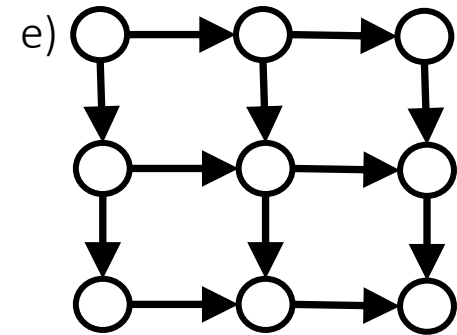
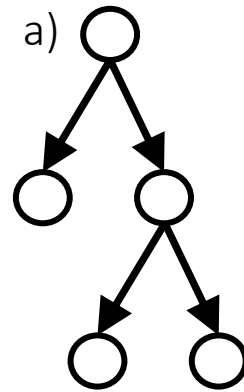
Naturalne
odzworowanie
złożonych struktur
i relacji świata
rzeczywistego.

Bazy grafowe należą do klasy baz **NoSQL**

Wybrane formy grafów

Przykładowe formy grafów

- a) Drzewo np.: struktura katalogów w systemie plików
- b) Gwiazda np.: komputerowa sieć lokalna
- c) Płatek śniegu np.: organizacja Państwa (województwa, powiaty, gminy, miasta...)
- d) Łańcuch np.: rzeka
- e) Siatka np.: sieć uliczna



Elementy baz grafowych

Węzły (Nodes)

Reprezentują obiekty takie jak osoby, miejsca, wydarzenia.

Krawędzie (Edges, Relationships)

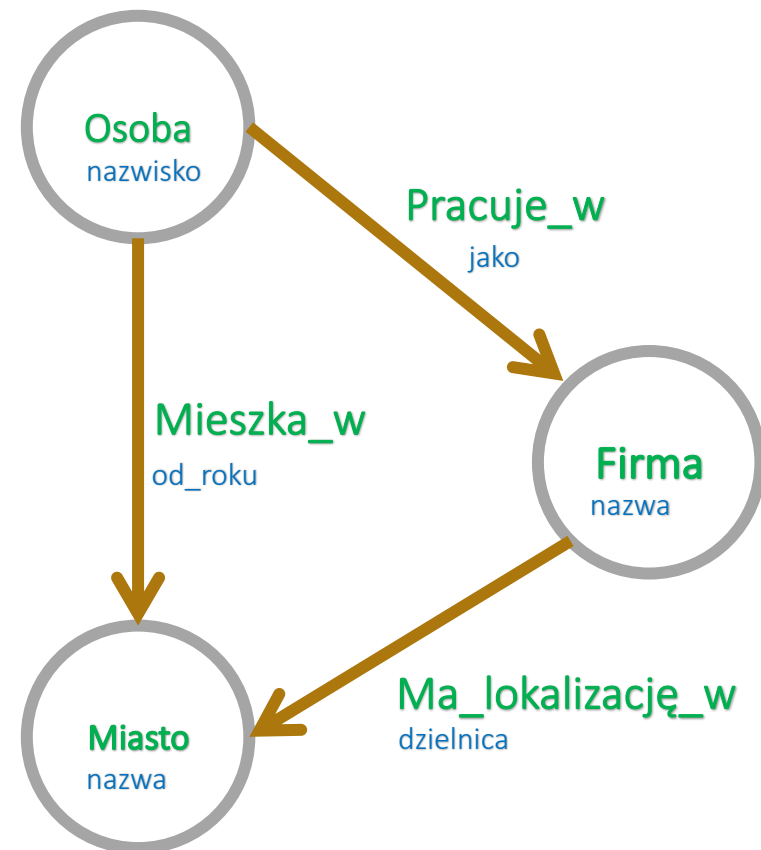
Reprezentują relacje między obiektami, np. znajomość, przepływ informacji.

Etykiety (Labels)

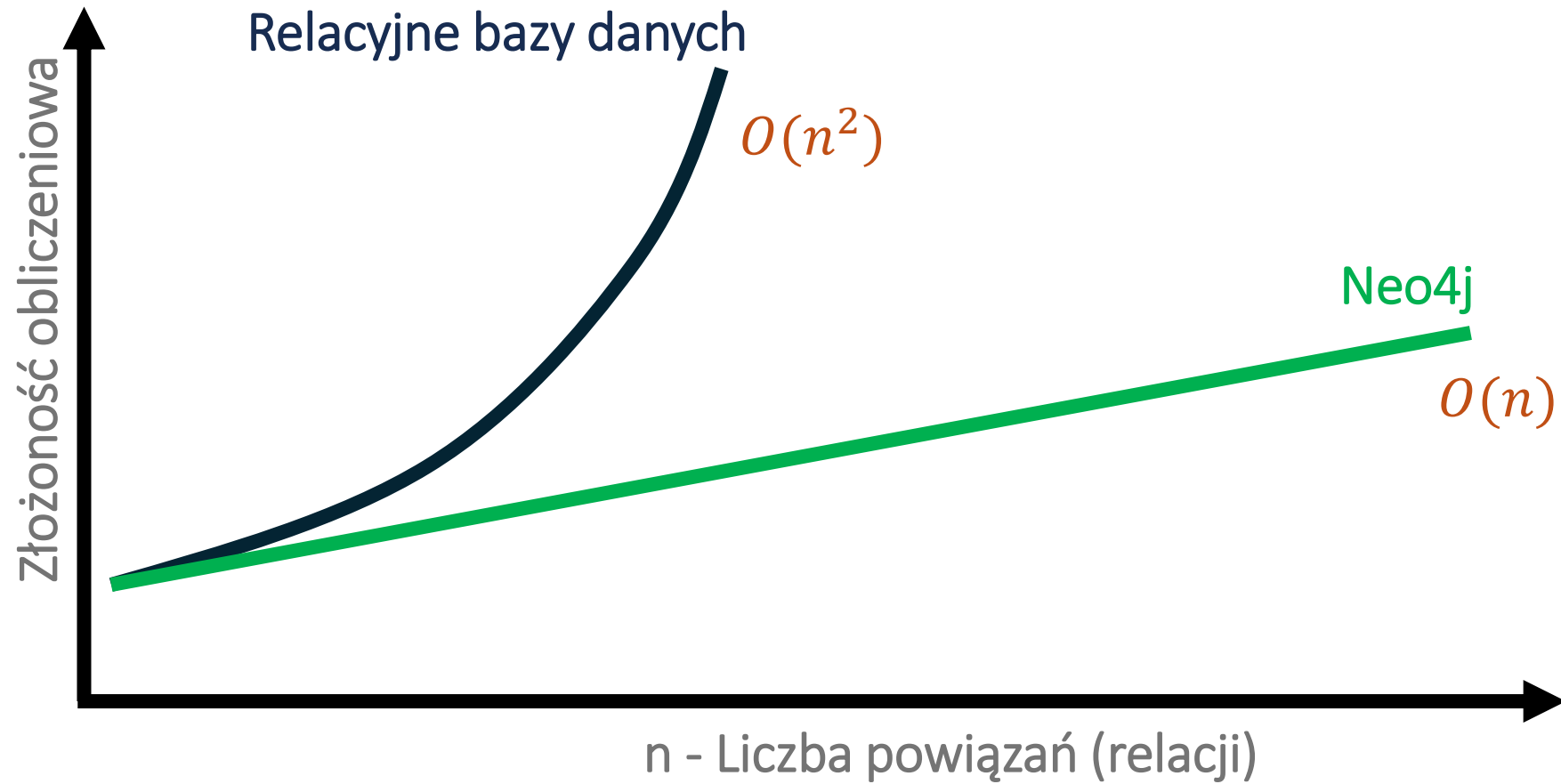
Klasyfikują węzły i krawędzie, ułatwiając organizację i wyszukiwanie danych.

Właściwości (Properties)

Informacje dodatkowe przypisane do węzłów i krawędzi, np. nazwa, dzielnica, rozmiar itd.



Bazy grafowe vs. Bazy relacyjne



Bazy grafowe vs. Bazy relacyjne

Relacyjna baza danych

Osoba

id	nazwisko
1	Nowak
2	Kowalski

Firma

id	Nazwa
1	Alfa
2	Beta

Miasto

id	nazwa
1	Warszawa
2	Kraków

Mieszka_w

idosoba	idmiasto	od_roku
1	2	1990
2	1	2011

Pracuje_w

idosoba	idfirma	jako
1	2	Mechanik
2	1	Księgowy

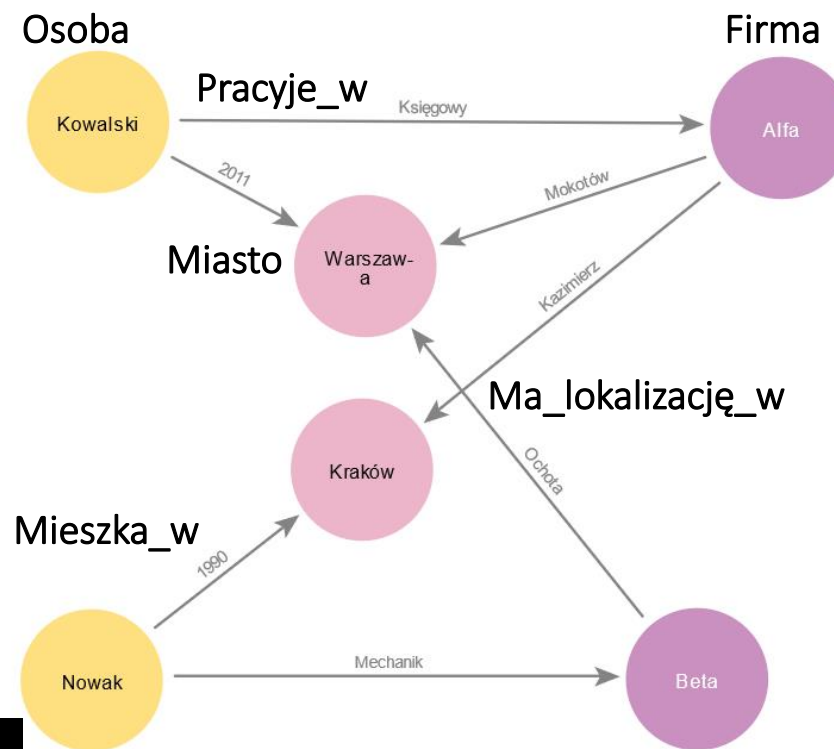
Ma_lokalizację_w

idfirma	idmiasto	dzielnica
1	1	Mokotów
1	2	Kazimierz
2	1	Ochota

SQL

```
SELECT Firma.nazwa, Pracuje_w.jako FROM Firma
JOIN Pracuje_w ON Pracuje_w.idfirma = Firma.id
JOIN Osoba ON Osoba.id = Pracuje_w.idosoba
WHERE Osoba.nazwisko = "Kowalski"
```

Grafowa baza danych



Cypher

```
MATCH (o:Osoba {nazwisko:'Kowalski'})-[p:Pracuje_w]->(f:Firma)
RETURN f.nazwa, p.jako
```

nazwa	jako
Alfa	Księgowy

Cypher - Relacyjne Zapytania

Cypher

to język zapytań deklaratywny, stworzony przez Neo4j dla baz danych grafowych. Umożliwia on efektywne tworzenie, modyfikowanie oraz przeszukiwanie grafów w bazie danych Neo4j.

Cypher skupia się na tym, co chcemy osiągnąć, a nie na tym, jak to osiągnąć

```
MATCH (o:Osoba)-[p:PRACUJE_DLA]->(f:Firma)
WHERE f.nazwa = 'Alfa' AND o.wiek > 30
RETURN o.imie, o.wiek, f.nazwa
ORDER BY o.wiek DESC LIMIT 10
```

CREATE, MERGE

- Tworzenie danych

MATCH

- Wyszukiwanie danych

SET, REMOVE

- Aktualizacja danych

DELETE

- Usuwanie danych

WITH, RETURN

- Kontrola przepływu

Porównanie Cypher z SQL

Powiązanie **zadań** z **procesem** i realizującymi je **osobami** pracującymi na **stanowiskach** w **komórkach** wewnętrznych organizacji

Cypher

```
MATCH d=(n:Proces {nazwa: 'Proces A'})-[:ZAWIERA_ZADANIE]->()-[:REALIZOWANE_PRZEZ]->()-[:PRACUJE_JAKO]->()-[:JEST_W]->(k)
RETURN d
```

SQL

```
SELECT * FROM Proces
INNER JOIN Zadanie ON Proces.id = Zadanie.idProces
INNER JOIN Osoba ON Zadanie.idOsoba = Osoba.id
INNER JOIN Stanowisko ON Osoba.idStanowisko = Stanowisko.id
INNER JOIN Komorka ON Stanowisko.idKomorka = Komorka.id
WHERE Proces.nazwa = 'Proces A';
```

Warunek, związki tylko 1:n

W przypadku związki n:n należało by dodać
- 4 tabele intersekcji
- 4 linie INNER JOIN



Algorytmy grafowe



Algorytmy wyszukiwania ścieżek

Wyszukują optymalne ścieżki pomiędzy węzłami dla określonych kryteriów



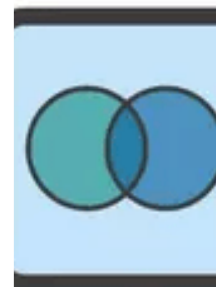
Algorytmy centralności

Pozwalają ocenić, jakie węzły w sieci są najważniejsze, czyli mogą mieć największy wpływ



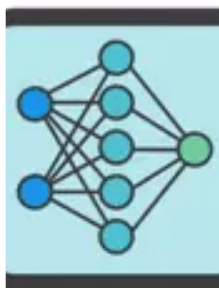
Algorytmy wspólnoty

Identyfikują skupiska węzłów, które są gęściej połączone między sobą niż z resztą sieci



Algorytmy podobieństwa

Obliczają stopień podobieństwa między elementami



Algorytmy detekcji anomalii

Wykorzystywane są do identyfikowania nieoczekiwanych wzorców lub obserwacji w danych



Algorytmy predykcji relacji

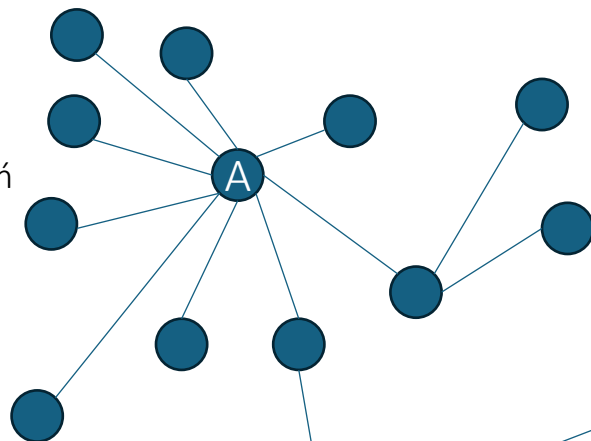
Prognozują czy dwa węzły w grafie mogą w przyszłości stworzyć połączenie

Algorytmy centralności

Degree Centrality
(Centralność stopnia)

Liczba połączeń?

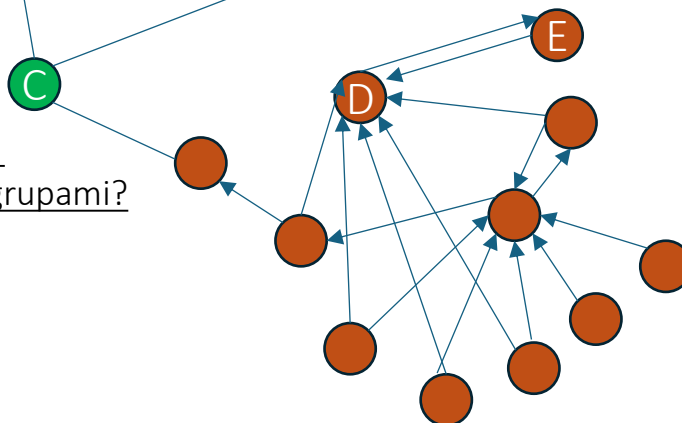
"A" ma najwyższy stopień



Betweenness Centrality
(Centralność bliskości)

Który węzeł ma największą kontrolę nad przepływem między węzłami i grupami?

"C" jest mostem



Louvain Modularity
(Modułowość Louvain)

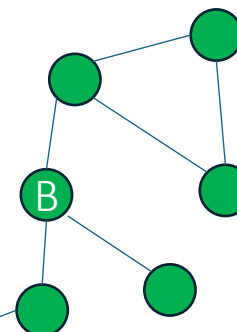
Jak efektywnie podzielić graf na społeczności?



Closeness Centrality
(Centralność pośrednictwa)

Który węzeł może najłatwiej dotrzeć do wszystkich innych węzłów w grafie?

"B" jest najbliższy z najmniejszą liczbą połączeń

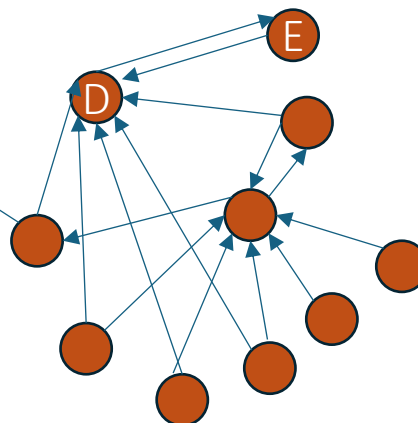


PageRanking

Który węzeł jest najważniejszy?

"D" jest na czele na podstawie liczby i wagi linków przychodzących

"E" jest następny, ze względu na wpływ linku "D"

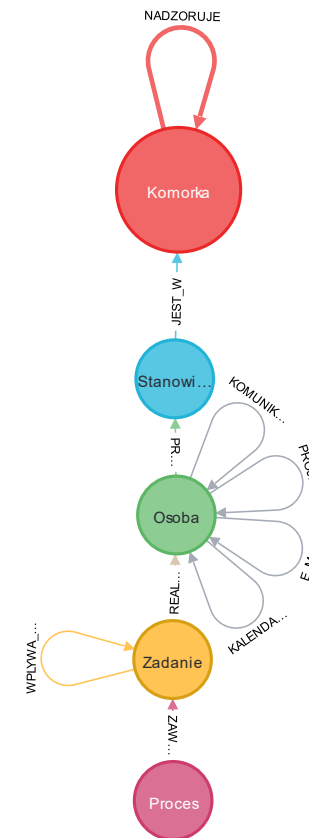
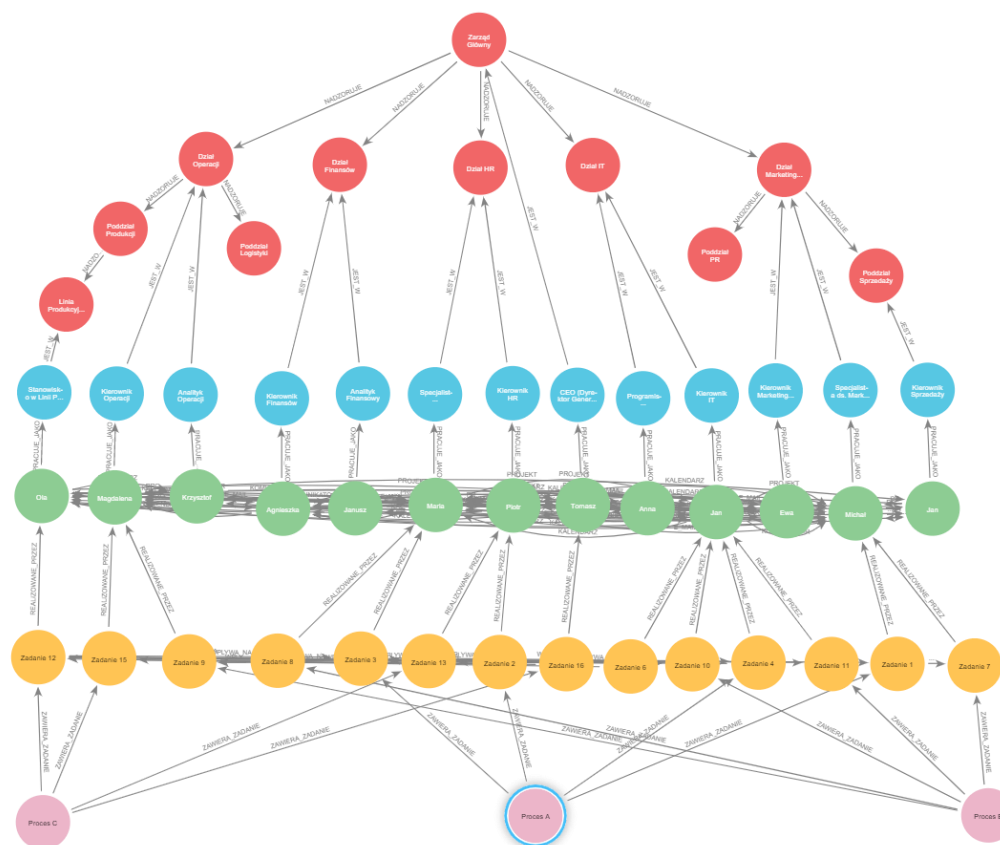


Publikacja



Wykorzystanie nowoczesnych metod z obszaru inżynierii w postaci baz grafowych w zarządzaniu organizacją

<https://www.drmalinowski.edu.pl/publications/39843-wykorzystanie-nowoczesnych-metod-z-obszaru-inzynierii-w-postaci-baz-grafowych-w-zarzadzaniu-organizacja>



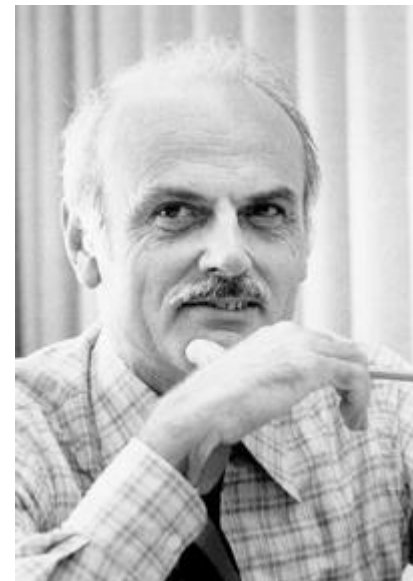
Model relacyjny

Edgar Frank Codd

Edgar Frank „Ted” Codd (ur. 23 sierpnia 1923 w Isle of Portland, zm. 18 kwietnia 2003 w Williams Island, Aventura) – brytyjski informatyk, wniósł wkład w rozwój teorii relacyjnych baz danych.

W 1970 roku wydał fundamentalną pracę ***A Relational Model of Data for Large Shared Data Banks***, w której przedstawił relacyjny model zarządzania bazami danych.

Ku jego rozczarowaniu IBM (gdzie pracował) zwlekał z wdrożeniem zaprezentowanych rozwiązań aż do chwili, gdy konkurencyjne firmy zaczęły stosować ten model – przykładem była baza danych Oracle, opracowana przez Larry’ego Ellisona na bazie idei Codda



Model relacyjny

Tabele (Tables)

Dane w bazach relacyjnych są przechowywane w tabelach, które składają się z wierszy (rekordów) i kolumn (atrybutów). Każda tabela reprezentuje zbiór podobnych obiektów, np. klientów, produktów czy zamówień.

Rekordy (Records)

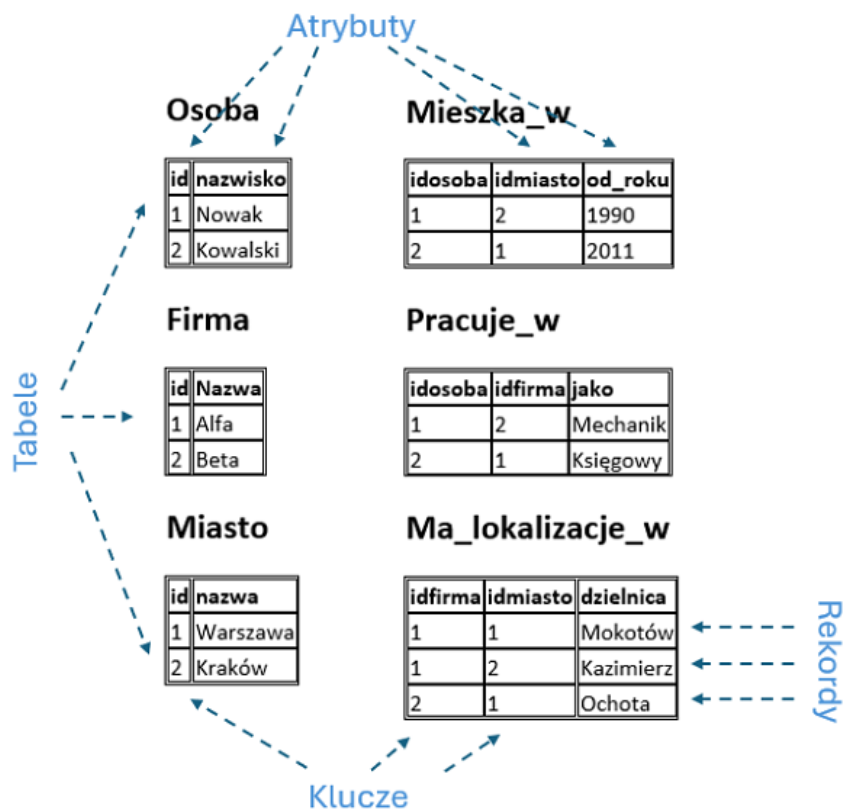
Każdy wiersz w tabeli nazywany jest rekordem i reprezentuje pojedynczy obiekt lub jednostkę danych. Na przykład, w tabeli "Klienci" każdy rekord może odpowiadać jednemu klientowi.

Atrybuty (Attributes)

Kolumny w tabeli określają atrybuty obiektów przechowywanych w tabeli. Na przykład, tabela "Klienci" może mieć kolumny takie jak "ID", "Imię", "Nazwisko" i "Adres".

Model relacyjny

Elementy



Relacje (Relationships)

Bazy relacyjne umożliwiają definiowanie relacji między tabelami poprzez klucze.

Relacje te mogą być typu jeden-do-jednego, jeden-do-wielu lub wiele-do-wielu, co pozwala na powiązanie danych z różnych tabel.

Klucze (Keys)

- Klucz główny (Primary Key) to unikalny identyfikator każdego rekordu w tabeli.
- Klucze obce (Foreign Keys) są używane do tworzenia relacji między tabelami, umożliwiając łączenie danych z różnych tabel.

Transakcja

Transakcja

to sekwencja operacji na bazie danych, traktowana jako jedna, niepodzielna jednostka pracy. Transakcja musi być wykonana w całości (commit) lub w ogóle (rollback), aby zapewnić spójność danych oraz musi spełniać właściwości ACID:

Atomicity (Atomowość):

- Wszystkie operacje w ramach transakcji są wykonane w całości lub żadna z nich. Jeśli jedna operacja się nie powiedzie, cała transakcja jest cofana.

Consistency (Spójność):

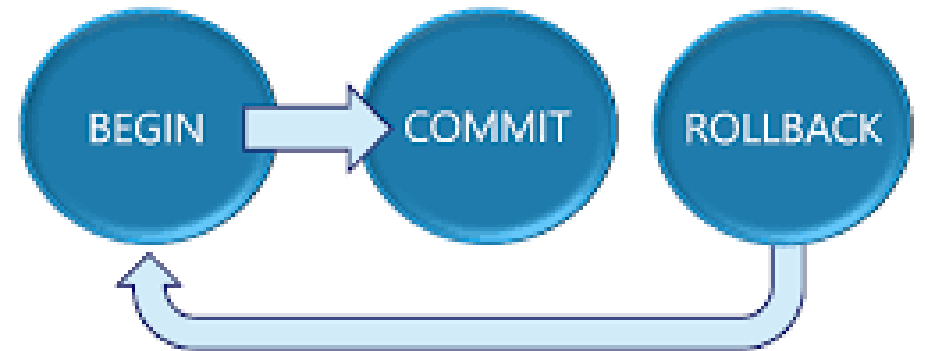
- Transakcja przekształca bazę danych z jednego poprawnego stanu do innego, zachowując przy tym wszystkie reguły integralności.

Isolation (Izolacja):

- Wyniki transakcji są niewidoczne dla innych transakcji, dopóki nie zostaną zatwierdzone, co zapobiega zakłóceniom między współbieżnymi transakcjami.

Durability (Trwałość):

- Po zatwierdzeniu transakcji, jej wyniki są trwale zapisane w bazie danych, nawet w przypadku awarii systemu.



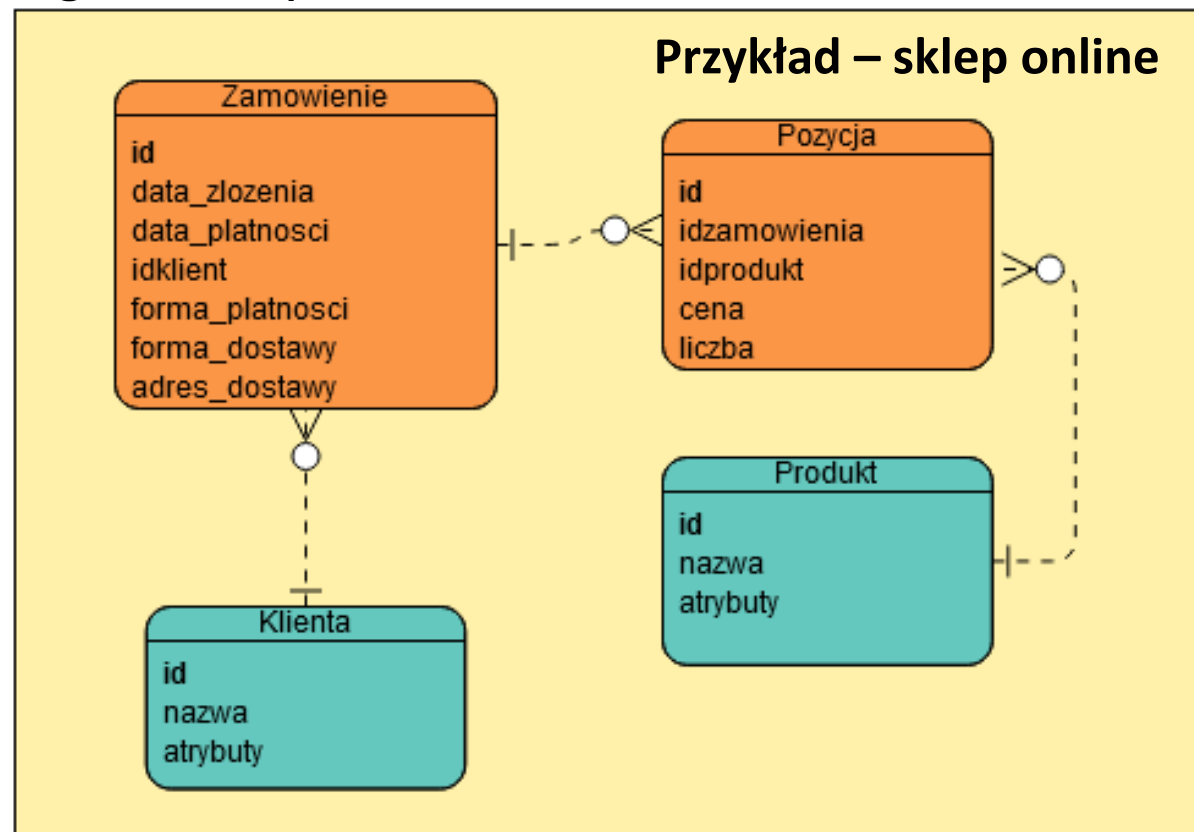
Operacje na relacjach

Operacja	Opis	Przykład
Selekcja (SELECT)	Wybiera wiersze z tabeli, które spełniają określone warunki.	Wybranie wszystkich klientów z miasta Warszawa.
Projekcja (PROJECT)	Wybiera określone kolumny z tabeli, eliminując duplikaty wierszy.	Wyświetlenie tylko nazwisk i adresów klientów.
Łączenie (JOIN)	Łączy dane z dwóch lub więcej tabel na podstawie wspólnego atrybutu.	Powiązanie danych o zamówieniach z danymi o klientach.
Różnica (DIFFERENCE)	Zwraca różnicę między dwiema tabelami, wybierając wiersze z jednej tabeli, których nie ma w drugiej.	Znalezienie klientów, którzy nie złożyli żadnego zamówienia.
Suma (UNION)	Łączy dane z dwóch tabel, zwracając wszystkie wiersze, eliminując duplikaty.	Połączenie listy klientów z różnych oddziałów w jedną listę.
Iloczyn Kartezjański (CARTESIAN PRODUCT)	Tworzy wszystkie możliwe kombinacje wierszy z dwóch tabel.	Połączenie każdej kombinacji produktów z każdym zamówieniem (zwykle z filtrem).

Diagram związków encji ERD

Diagram związków encji (Entity-Relationship Diagram, ERD)

to graficzna reprezentacja struktury bazy danych. Jest to narzędzie używane do modelowania i analizy systemów baz danych. ERD pomaga w zrozumieniu relacji pomiędzy różnymi encjami (takimi jak tabele w bazach danych) oraz atrybutami tych encji (takimi jak pola w tabelach).



Język SQL

SQL (Structured Query Language)

to strukturalny język zapytań, który jest standardem w zarządzaniu i manipulacji danymi w relacyjnych bazach danych. Umożliwia tworzenie, modyfikowanie, odczytywanie oraz zarządzanie danymi przechowywanymi w tabelach, które są połączone relacjami.

Cechy

Deklaratywność:

- Jest językiem deklaratywnym, co oznacza, że użytkownik określa, co chce osiągnąć, ale nie musi wskazywać, jak to zrobić.

Spójność i integralność danych:

- Pozwala na zachowanie integralności danych, zapobiegając duplikatom oraz błędom związanym z relacjami między danymi.

Obsługa transakcji:

- Umożliwia zarządzanie transakcjami poprzez mechanizmy ACID (Atomicity, Consistency, Isolation, Durability)

Obsługa wielu baz danych:

- Jest standardem wspieranym przez większość relacyjnych systemów baz danych

Język SQL

Grupy komend

DQL (Data Query Language)	DDL (Data Definition Language)	DML (Data Manipulation Language)	DCL (Data Control Language)	TCL (Transaction Control Language)
Pobieranie danych z bazy danych.	Definiowanie i modyfikowanie struktury bazy danych.	Manipulowanie danymi w tabelach.	Zarządzanie uprawnieniami dostępu do danych.	Zarządzanie transakcjami w bazie danych.
SELECT, WHERE, INNER	CREATE, ALTER, DROP	INSERT, UPDATE, DELETE	GRANT, REVOKE	COMMIT, ROLLBACK, SAVEPOINT

Język SQL

Tabela 1: Klienci

KlientID	Imie	Nazwisko	Miasto
1	Jan	Nowak	Warszawa
2	Anna	Kowalska	Kraków
3	Piotr	Wiśniewski	Gdańsk
4	Ewa	Kwiatkowska	Warszawa
5	Marek	Kowalski	Wrocław

```
SELECT Imie, Nazwisko
FROM Klienci;
```

Imie	Nazwisko
Jan	Nowak
Anna	Kowalska
Piotr	Wiśniewski
Ewa	Kwiatkowska
Marek	Kowalski

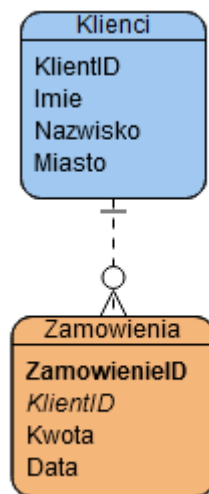


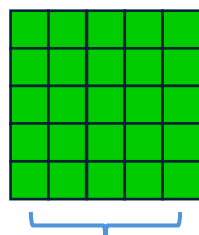
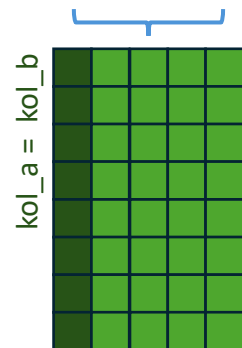
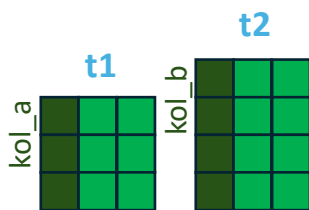
Tabela 2: Zamowienia

ZamowienieID	KlientID	Kwota	Data
101	1	1500	10.01.2023
102	2	2500	15.02.2023
103	1	2000	05.03.2023
104	3	3000	01.04.2023
105	4	1200	12.05.2023
106	4	5000	01.06.2023

```
SELECT Klienci.Imie, Klienci.Nazwisko, Zamowienia.Kwota
FROM Klienci
INNER JOIN Zamowienia ON Klienci.KlientID = Zamowienia.KlientID;
```

Imie	Nazwisko	Kwota
Jan	Nowak	1500
Jan	Nowak	2000
Anna	Kowalska	2500
Piotr	Wiśniewski	3000
Ewa	Kwiatkowska	1200
Ewa	Kwiatkowska	5000

SQL - SELECT



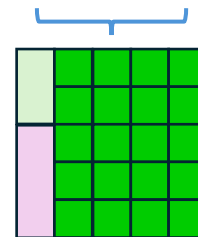
SELECT kol_a, kol_b

FROM t1

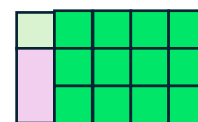
JOIN t2

ON t1.kol_a = t2.kol_b

WHERE warunki



GROUP BY kol



HAVING warunki



ORDER BY kol ASC/DESC



LIMIT liczba



Hurtownie danych

Hurtownia danych

to centralny, zintegrowany system do przechowywania dużych ilości danych pochodzących z różnych źródeł.

Jest wykorzystywany głównie do celów analitycznych i raportowania, umożliwiając organizacjom analizowanie historycznych i bieżących danych, wspierając podejmowanie decyzji na poziomie strategicznym.

Cechy

Zintegrowane źródło danych:

Konsoliduje informacje z różnych systemów (np. systemy CRM, ERP, e-commerce) w jednym miejscu, zapewniając jednolitą strukturę danych.

Tematyczne przechowywanie danych:

Dane są organizowane wokół głównych tematów biznesowych, takich jak sprzedaż, finanse, logistyka czy zasoby ludzkie.

Długoterminowe przechowywanie danych:

Przechowują dane historyczne przez dłuższy czas, umożliwiając analizowanie trendów i wzorców w oparciu o dane sprzed wielu lat.

Stołość danych:

Są zwykle niezmiennie. Po załadowaniu do hurtowni rzadko się je aktualizuje..

Dane zorientowane na użytkownika:

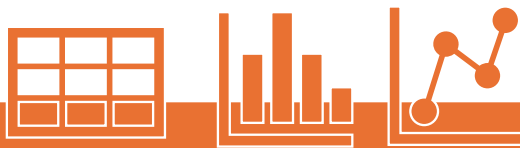
Jest optymalizowana pod kątem zapytań analitycznych (De-normalizacja).

Hurtownie danych

Architektura

Warstwa źródłowa (Źródła danych):

- Różnorodne systemy, z których pobierane są dane, np. bazy danych, systemy transakcyjne, pliki CSV, dane z API, logi systemowe. Dane są pozyskiwane z wielu formatów i źródeł.



Proces ETL (Extract, Transform, Load):

- Kluczowy element hurtowni danych, w którym dane są przetwarzane:



Warstwa hurtowni danych:

- Główna baza danych, w której przechowywane są dane, zoptymalizowana do odczytu. Dane są zazwyczaj organizowane w formie schematów gwiazdy lub płatka śniegu, co umożliwia szybkie zapytania analityczne.



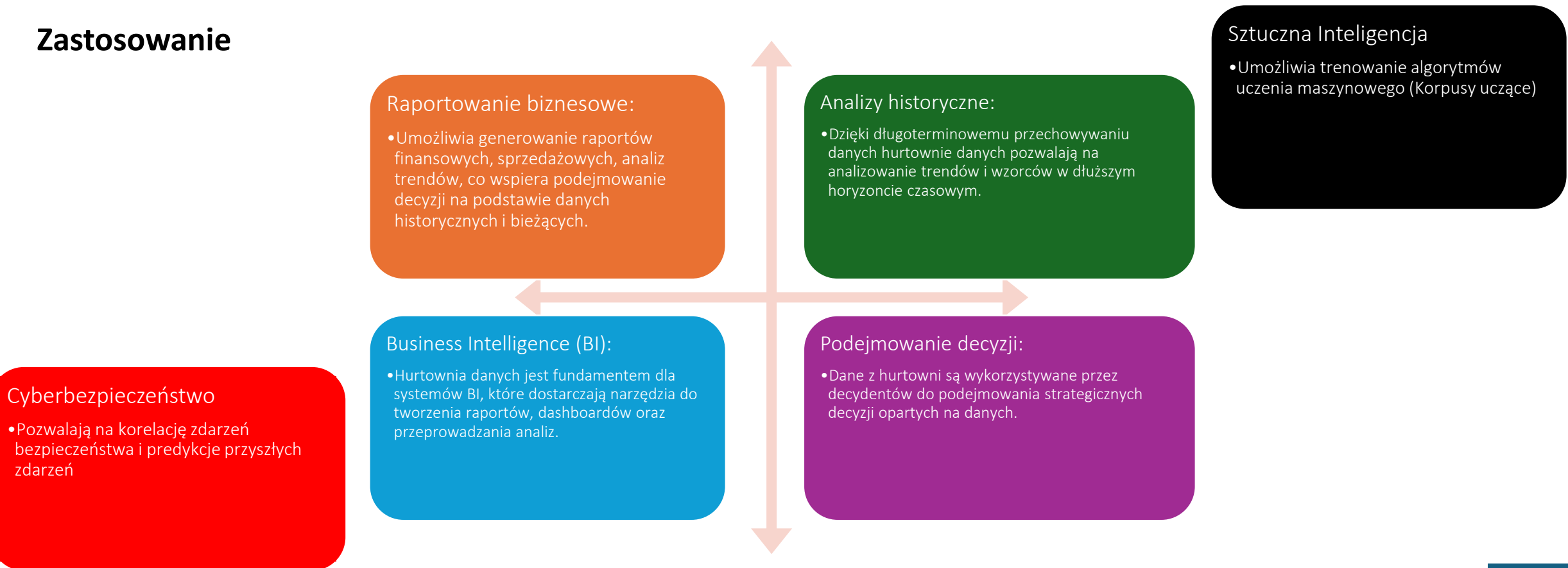
Warstwa dostępu do danych:

- Użytkownicy końcowi, analitycy, systemy raportowania i narzędzia BI (Business Intelligence) uzyskują dostęp do danych z hurtowni za pośrednictwem interfejsów zapytań, raportów lub dashboardów.



Hurtownie danych

Zastosowanie



Hurtownie danych

ETL vs ELT

ETL (Extract, Transform, Load)

ETL to tradycyjny proces przetwarzania danych, który składa się z trzech głównych etapów:

1. **Extract (Ekstrakcja):** Dane są pobierane ze źródeł, takich jak bazy danych, pliki czy aplikacje.
2. **Transform (Transformacja):** Dane są przekształcane, czyszczone, łączone i formatowane zgodnie z wymaganiami docelowej bazy danych lub aplikacji.
3. **Load (Ładowanie):** Przekształcone dane są ładowane do docelowej hurtowni danych lub bazy danych.

ELT (Extract, Load, Transform)

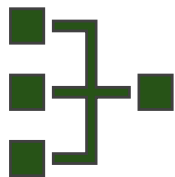
ELT to nowsze podejście do przetwarzania danych, które również składa się z trzech etapów, ale różni się kolejnością:

1. **Extract (Ekstrakcja):** Dane są pobierane ze źródeł, podobnie jak w ETL.
2. **Load (Ładowanie):** Surowe dane są ładowane bezpośrednio do docelowej hurtowni danych lub bazy danych.
3. **Transform (Transformacja):** Transformacja danych odbywa się po ich załadowaniu, zazwyczaj przy użyciu mocy obliczeniowej samej hurtowni danych lub systemu analitycznego.

Hurtownie danych

Źródła surowych danych

ETL



Extract



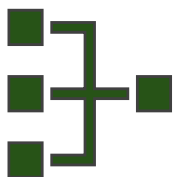
Transform

Load



Docelowa baza danych

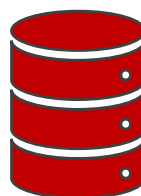
ELT



Extract



Load



Transform

Wybierz ETL, gdy:

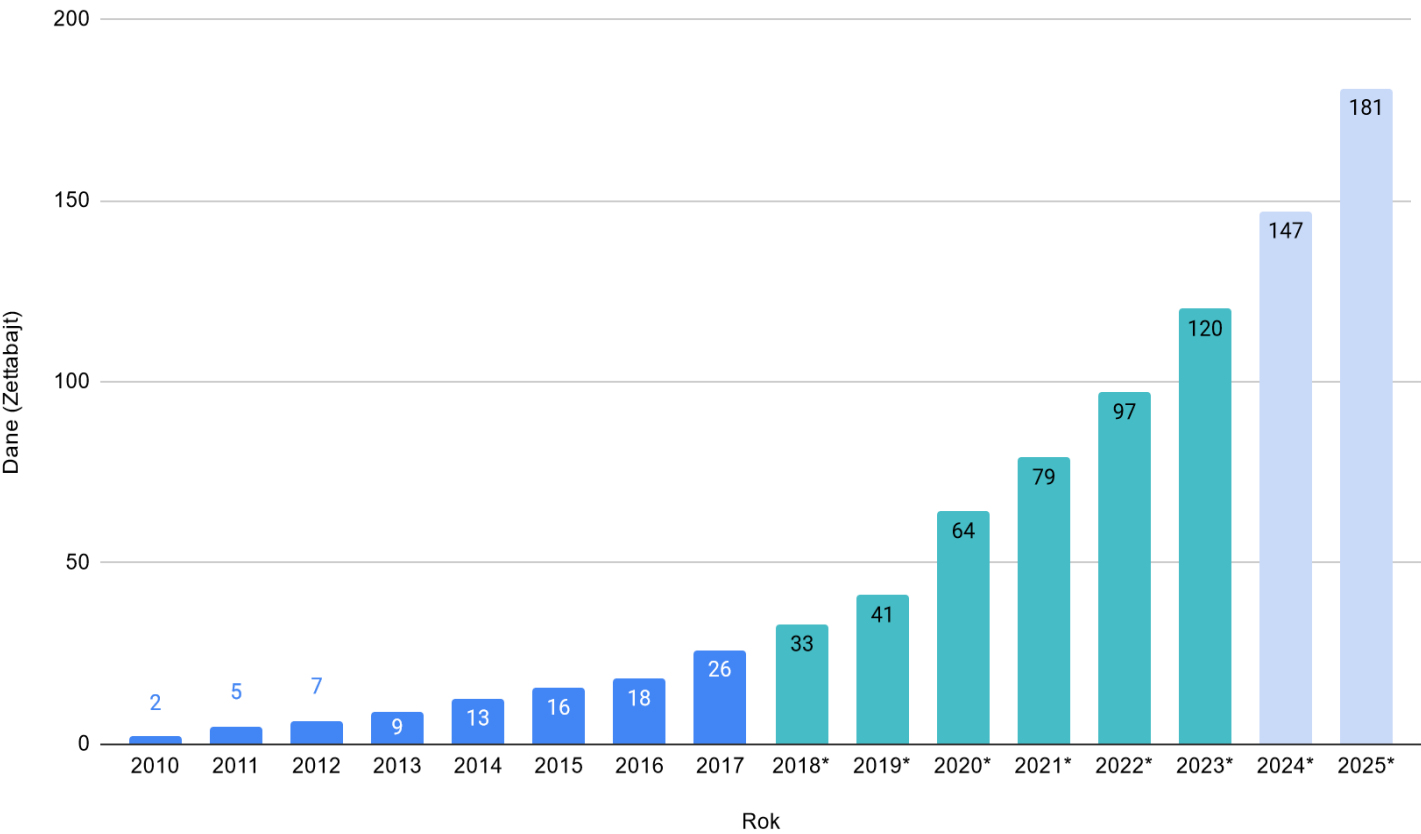
- Pracujesz z tradycyjnymi systemami hurtowni danych.
- Jakość i integralność danych jest priorytetem.
- Transformacja danych jest złożona i wymaga dokładnej kontroli przed załadowaniem.

Wybierz ELT, gdy:

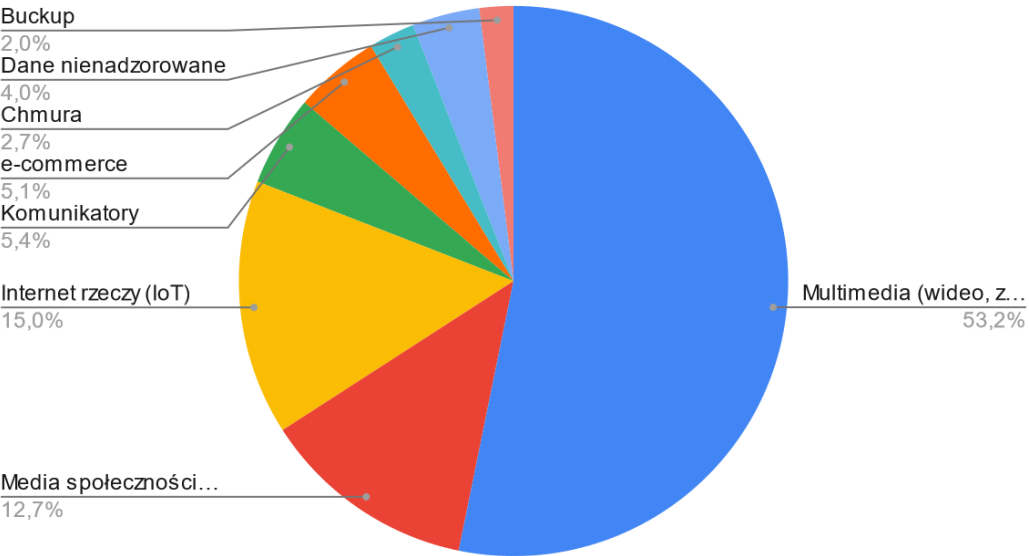
- Pracujesz z dużymi zestawami danych w środowisku Big Data lub chmurowym.
- Chcesz szybko ładować dane i wykonywać transformacje w systemie docelowym.
- Masz dostęp do dużej mocy obliczeniowej, która umożliwia efektywną transformację po załadowaniu danych.

Big Data

Wielkość danych w latach na Świecie



Typy danych



Na osobę	
2022*	12 TB
2023*	15 TB
2024*	18 TB
2025*	23 TB

Big Data

Big Data

to termin odnoszący się do ogromnych i złożonych zbiorów danych, które są zbyt duże, aby mogły być przetwarzane przez tradycyjne systemy baz danych.

Dane te charakteryzują się dużą objętością, różnorodnością i szybkością przetwarzania, co wymaga specjalistycznych technologii oraz narzędzi do analizy i zarządzania nimi.

Główne cechy (5 V)

Volume (Objętość):

- Ogromne ilości danych generowanych w krótkim czasie, często liczone w petabajtach lub eksabajtach.

Variety (Różnorodność):

- Dane dostępne są w różnych formatach, zarówno ustrukturyzowanych, półustrukturyzowanych, jak i nieustrukturyzowanych.

Velocity (Szybkość):

- Dane są generowane i przetwarzane w dużym tempie, często w czasie rzeczywistym, aby uzyskać natychmiastowe wyniki.

Veracity (Prawdziwość):

- Kluczowa jest jakość i wiarygodność danych, które muszą być dokładne, aby przyniosły wartościowe rezultaty.

Value (Wartość):

- Dane mają wartość tylko wtedy, gdy są odpowiednio przetworzone i zanalizowane, aby dostarczać użyteczne informacje wspierające decyzje.

Big Data

Zastosowanie

Marketing i personalizacja:

- Firmy mogą analizować dane dotyczące zachowań użytkowników online, aby dostosować reklamy i oferty do indywidualnych preferencji klientów.

Analiza predykcyjna:

- W branży finansowej i ubezpieczeniowej Big Data jest wykorzystywana do analizy ryzyka, przewidywania zmian rynkowych, a także w detekcji oszustw.

Zdrowie i medycyna:

- Analiza dużych zbiorów danych zdrowotnych pomaga w diagnozowaniu chorób, monitorowaniu pacjentów w czasie rzeczywistym oraz w rozwoju nowych terapii i leków.

IoT (Internet of Things):

- Urządzenia IoT generują ogromne ilości danych, które są analizowane w czasie rzeczywistym, aby zarządzać i optymalizować infrastrukturą, np. inteligentne miasta, systemy transportowe.

Media społecznościowe:

- Analiza danych z mediów społecznościowych pozwala na monitorowanie trendów, nastrojów konsumenckich, a także identyfikację wpływowych użytkowników.

Big Data

Big Data vs Hurtownia danych

Kryterium	Big Data	Hurtownia Danych
Rodzaj danych	Ustrukturyzowane, półustrukturyzowane, nieustrukturyzowane (np. dane tekstowe, multimedia).	Głównie ustrukturyzowane dane w tabelach relacyjnych.
Cel	Analiza dużych, złożonych zbiorów danych, analiza predykcyjna, wykrywanie wzorców i trendów.	Raportowanie i analiza historyczna, wspieranie decyzji biznesowych.
Skalowalność i rozmiar	Obsługuje ogromne ilości danych (petabajty i więcej), dane generowane w czasie rzeczywistym.	Obsługuje duże, ale ograniczone rozmiary danych, zazwyczaj z systemów informacyjnych organizacji.
Technologie	Hadoop, Apache Spark, NoSQL, chmura obliczeniowa.	RDBMS (Relacyjne Systemy Zarządzania Bazą Danych), ETL (Extract, Transform, Load).
Przetwarzanie danych	Przetwarzanie wsadowe i strumieniowe w czasie rzeczywistym.	Przetwarzanie wsadowe z okresowymi aktualizacjami (np. co dzień, tydzień).
Struktura danych	Złożone, dynamiczne, niejednolite dane.	Dane ustrukturyzowane.

