

Internetowe bazy danych

Projektowanie i programowanie systemów internetowych I

mgr inż. Krzysztof Rewak

16 kwietnia 2018

Wydział Nauk Technicznych i Ekonomicznych

Państwowa Wyższa Szkoła Zawodowa im. Witelona w Legnicy

Plan prezentacji

1. `SELECT * FROM definitions;`
2. `SELECT * FROM models;`
3. `SELECT * FROM models WHERE name = "flat";`
4. `SELECT * FROM models WHERE name = "hierarchical";`
5. `SELECT * FROM models WHERE name = "relational";`
6. `SELECT * FROM models WHERE name = "object-oriented";`
7. `SELECT * FROM models WHERE name = "nosql";`
8. Podsumowanie

SELECT * FROM definitions;

Czym są dane?

W ujęciu informatycznym jest to sekwencja symboli, która po pewnej interpretacji może zostać uznana za informację.

Czym są w takim razie bazy danych?

Uporządkowane w pewien sposób zbiory danych.

DBMS?

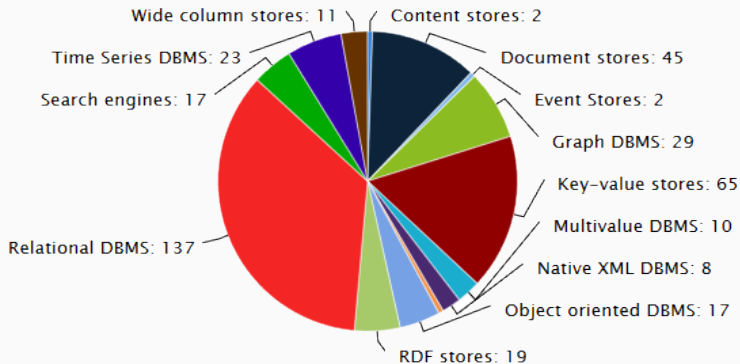
Jest to system zarządzania bazą danych (ang. *Database Management System*), czyli system pozwalający między innymi na pobieranie i modyfikowanie danych w bazie danych.

SELECT * FROM models;

Modele baz danych możemy podzielić na kilka typów:

- NoSQL
 - klucz/wartość
 - dokumentowe (?)
 - grafowe
- obiektowe
- relacyjne
- hierarchiczne
- kartotekowe
- oraz wiele innych

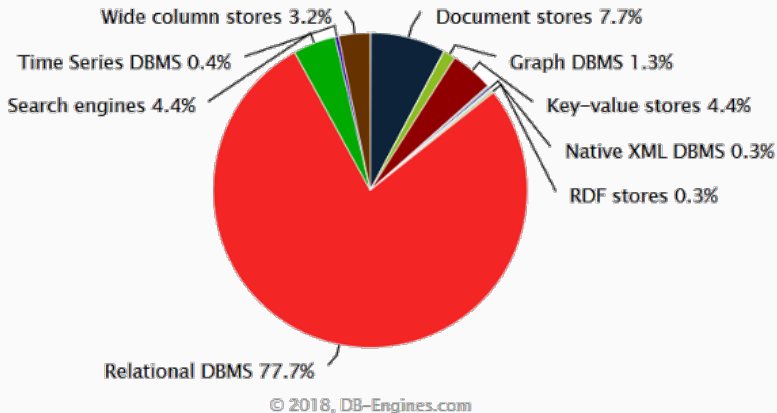
Dla odmiany zaczniemy omawianie od końca listy.



© 2018, DB-Engines.com

Rysunek 1: Liczba DBMS wg kategorii wg

https://db-engines.com/en/ranking_categories



Rysunek 2: Popularność DBMS wg kategorii wg
https://db-engines.com/en/ranking_categories

```
SELECT * FROM models  
WHERE name = "flat";
```

Kartotekowe modele bazodanowe

year;	month;	income;	gain;	currency;
2018;	01;	567432.43;	243854.04;	PLN;
2018;	02;	785395.28;	387104.54;	PLN;
2018;	03;	599482.41;	295859.16;	PLN;
2018;	04;	257179.91;	104950.32;	PLN;

Co można zrobić?

- presortować,
- przeszukać.

Czego nie można zrobić?

- zazwyczaj wymusić typu przechowywanych danych;
- połączyć z inną bazą danych;
- korzystać z indeksów, kluczy.

Zalety?

- prostota obsługi,
- często *system-agnostic*,
- zrozumiałe nawet po bezpośrednim otwarciu.

Jak można utworzyć?

- w edytorze tekstu,
- w arkuszu kalkulacyjnym,
- w innym programie do tego przeznaczonym.

```
SELECT * FROM models  
WHERE name = "hierarchical";
```

Hierarchiczne modele bazodanowe

authors:

id; name;

1; Richard K. Morgan

2; Walter Jon Willians

3; Dan Simmons

books:

id; title; author_id

1; Praxis; 2

2; Rozpad; 2

3; Wojna; 2

4; Modyfikowany wegiel; 1

5; Upadle anioly; 1

6; Zbudzone furie; 1

7; Hyperion; 3

8; Upadek Hyperiona; 3

9; Endymion 3

10; Triumf Endymiona; 3

Hierarchiczne modele bazodanowe

Co można zrobić?

- połączyć rekordy z kilku tabel w drzewiastej strukturze,
- sortować po rekordach natywnych i zależnych,
- przeszukiwać po rekordach natywnych i zależnych.

Czego nie można zrobić?

- stworzyć relacji typu *ma wiele*,
- stworzyć relacji typu *wiele do wielu*.

Hierarchiczne modele bazodanowe

Zalety?

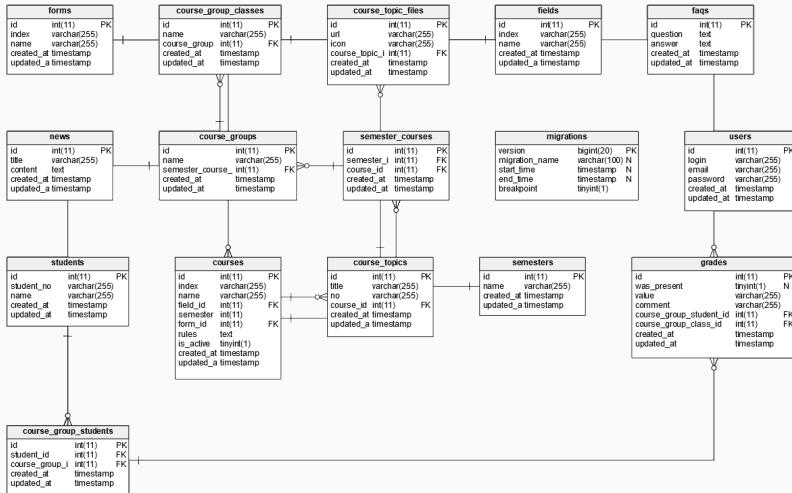
- prostota obsługi;
- czytelność schematu bazy;
- szybkie w porównaniu do klasycznej relacyjnej bazy danych.

Zastosowania?

- rejestr Windows,
- bankowość,
- telekomunikacja.

```
SELECT * FROM models  
WHERE name = "relational";
```

Relacyjne modele bazodanowe



Relacyjne modele bazodanowe

Co można zrobić?

- połączyć rekordy z kilku tabel w relacyjnej strukturze,
- sortować po rekordach natywnych i zależnych,
- przeszukiwać po rekordach natywnych i zależnych,
- ustanowić klucze obce, co może zabezpieczyć integralność danych,
- ustanowić indeksy, które pomogą przy przeszukiwaniu i sortowaniu tabel,
- relatywnie łatwo rozproszyć w pionie lub poziomie,
- korzystać z transakcji.

ACID, czyli

- *atomicity* - niepodzielność
- *consistency* - spójność
- *isolation* - izolacja
- *durability* - trwałość

Są to cechy gwarantujące poprawne przetwarzanie transakcji w (relacyjnych) bazach danych.

Relacyjne modele bazodanowe

Relacyjne bazy danych są obecnie najpopularniejszym modelem wykorzystywanym praktycznie wszędzie: od małych prywatnych projektów studenckich, przez produkty startupów i systemy obsługujące duże firmy, aż po korporacyjne rozwiązania.

Relacyjne modele bazodanowe

Rank	Apr 2018	Mar 2018	Apr 2017	DBMS	Database Model	Score		
						Apr 2018	Mar 2018	Apr 2017
1.	1.	1.		Oracle	Relational DBMS	1289.79	+0.18	-112.21
2.	2.	2.		MySQL	Relational DBMS	1226.40	-2.46	-138.22
3.	3.	3.		Microsoft SQL Server	Relational DBMS	1095.51	-9.28	-109.26
4.	4.	4.		PostgreSQL	Relational DBMS	395.47	-3.88	+33.69
5.	5.	5.		MongoDB	Document store	341.41	+0.89	+15.98
6.	6.	6.		DB2	Relational DBMS	188.95	+2.28	+2.29
7.	7.	7.		Microsoft Access	Relational DBMS	132.22	+0.27	+4.04
8.	9.	11.		Elasticsearch	Search engine	131.36	+2.81	+25.69
9.	8.	9.		Redis	Key-value store	130.11	-1.12	+15.75
10.	10.	8.		Cassandra	Wide column store	119.09	-4.40	-7.10
11.	11.	10.		SQLite	Relational DBMS	115.99	+1.17	+2.19
12.	12.	12.		Teradata	Relational DBMS	73.68	+1.21	-2.88
13.	13.	17.		Splunk	Search engine	65.06	-0.61	+9.55
14.	15.	18.		MariaDB	Relational DBMS	64.56	+1.45	+15.83
15.	14.	14.		Solr	Search engine	63.21	-1.60	-1.16
16.	16.	13.		SAP Adaptive Server	Relational DBMS	61.63	-0.99	-5.83
17.	17.	15.		HBase	Wide column store	59.69	-1.24	+1.22
18.	18.	20.		Hive	Relational DBMS	57.40	+0.39	+15.75
19.	19.	16.		FileMaker	Relational DBMS	55.00	-0.12	-2.17
20.	20.	19.		SAP HANA	Relational DBMS	48.90	+0.37	+0.75
21.	21.	22.		Amazon DynamoDB	Multi-model	43.14	+0.69	+11.08
22.	22.	21.		Neo4j	Graph DBMS	40.90	-0.00	+5.99

```
SELECT * FROM models  
WHERE name =  
"object-oriented";
```

Obiektowe modele bazodanowe

```
ObjectContainer oc = Db4o.openFile("database.db4o");

Author author = new Author("Cixin Liu");
Book book = new Book("Ciemny las", author);

oc.store(author);
oc.store(book);




oc.close();
```

Obiektowe modele bazodanowe

Jak działają?

- są zbiorem obiektów wedle definicji obiektu wykorzystywanego języka programowania z wszystkimi tego wadami i zaletami,
- są przetrzymywane w formie zserializowanej lub zapisanej w inny sposób, co likwiduje potrzebę mapowania danych,
- są wygodne w użyciu dla programistów znających dany język programowania,
- nie są szczególnie popularne przez konkurencję ze strony systemów ORM.

Obiektowe modele bazodanowe

	Rank			DBMS	Database Model	Score		
	Apr 2018	Mar 2018	Apr 2017			Apr 2018	Mar 2018	Apr 2017
94.	↑	100.	↑	96.	mSQL	Relational DBMS	1.89 +0.01	+0.14
95.	↓	90.	↓	76.	Amazon SimpleDB	Key-value store	1.89 -0.50	-0.84
96.	↓	92.	↓	90.	Percona Server for MySQL	Relational DBMS	1.85 -0.40	-0.03
97.	↑	102.	↓	91.	Virtuoso	Multi-model 	1.80 -0.02	-0.06
98.	↓	94.	↓	89.	CloudKit	Document store	1.74 -0.44	-0.18
99.	↓	96.	↓	97.	OpenTSDB	Time Series DBMS	1.70 -0.32	+0.18
100.	↓	97.	↓	73.	Teradata Aster	Relational DBMS	1.66 -0.28	-1.16
101.	↑	104.	↑	113.	RocksDB	Key-value store	1.57 -0.15	+0.56
102.	↓	98.	↓	94.	Datomic	Relational DBMS	1.51 -0.42	-0.29
103.		103.	↑	106.	jBASE	Multivalue DBMS	1.43 -0.39	+0.23
104.	↑	107.	↓	103.	VoltDB	Relational DBMS	1.40 -0.26	+0.08
105.	↑	108.	↓	99.	MonetDB 	Relational DBMS	1.35 -0.13	-0.14
106.	↓	101.	↓	84.	IBM dashDB	Relational DBMS	1.34 -0.49	-0.78
107.	↓	105.	↓	95.	IMS	Navigational DBMS	1.33 -0.36	-0.42
108.	↓	106.	↓	104.	EnterpriseDB 	Relational DBMS	1.32 -0.35	+0.05
109.	↑	115.	↓	107.	Red Brick	Relational DBMS	1.23 +0.08	+0.04
110.	↑	113.	↓	102.	Db4o	Object oriented DBMS	1.21 -0.01	-0.23
111.	↓	109.	↓	108.	GridGain	Key-value store	1.21 -0.24	+0.06
112.	↑	133.	↑	154.	Google Cloud Spanner	Relational DBMS	1.18 +0.22	+0.66
113.	↓	110.	↓	101.	Datameer	Document store	1.17 -0.20	-0.28
114.	↓	111.	↓	110.	Tibero	Relational DBMS	1.12 -0.21	+0.09
115.	↑	122.	↑	122.	Versant Object Database	Object oriented DBMS	1.08 +0.03	+0.19

Obiektowe modele bazodanowe

Dlaczego więc w ogóle warto o nich wspominać?

Przede wszystkim dlatego, że są wygodne w użyciu i zasada ich działania pokrywa się z zasadą działania systemów mapujących relacje na obiekty. Ale o tym na następnym wykładzie.

```
SELECT * FROM models  
WHERE name = "nosql";
```

Nierelacyjne modele bazodanowe

Opracowano wiele podejść do NoSQL-owych baz danych, a do najpopularniejszych należą:

- *key-value*
- *document-oriented*
- *graph*
- *column*
- *multi-model*

Nierelacyjne modele bazodanowe

Model klucz-wartość, często wykorzystywany jako cache, korzysta z mapy, słownika lub asocjacyjnej tablicy:

```
novel:54:author           // Walter Jon Williams
novel:54:originaltitle    // The Praxis
novel:54:pages            // 8
novel:54:readdatefrom     // 2018-02-07
novel:54:readdateto      // 2018-02-17
novel:54:readtitle       // Praxis
novel:54:releaseyear     // 2002
```

Nierelacyjne modele bazodanowe

Co jest istotne?

- każdy klucz może pojawić się tylko raz,
- wykorzystywany chociażby przy procesach zakupowych lub systemach cache,
- łatwo znaleźć dane po kluczu, a (zazwyczaj) niekoniecznie po ich wartości,
- idealne do skalowania.

Nierelacyjne modele bazodanowe

NoSQL-owe bazy mogą przetrzymywać ustrukturyzowane dokumenty:

```
{
  "author": {
    "firstname": "Walter Jon",
    "lastname": "Williams",
    "language": {
      "name": "angielski",
      "code": "en",
    }
  },
  "originaltitle": "The Praxis",
  "pages": 384,
  "readdatefrom": "2018-02-07",
  "readdateto": "2018-02-17",
  "readtitle": "Praxis",
  "releaseyear": "2002",
  "uuid": "a41b2bd7-1729-4c0d-b6a0-721ddad0a5ef",
}
```

Nierelacyjne modele bazodanowe

Co jest istotne?

- pliki mogą być zapisane jako JSON, XML lub dowolny innych format,
- każdy dokument powinien mieć swój unikalny identyfikator,
- ze względu na strukturę, dane mogą być bardzo elastycznie (nie)porządkowane,
- łatwo skalowalne i przeznaczone do przechowywania ogromnych ilości danych.

Nierelacyjne modele bazodanowe

Kolumnowe bazy składają się z plików, które przechowują po trzy zmienne: nazwę kolumny, jej wartość oraz znacznik czasowy:

```
{  
  {  
    name: "author",  
    value: "Walter Jon Williams",  
    timestamp: 1523819873  
  },  
  {  
    name: "pages",  
    value: 384,  
    timestamp: 1523819873  
  },  
}
```

Podsumowanie

HDD?

Najważniejsze jest zinterpretowanie potrzeb danego systemu i dobranie do nich odpowiedniego narzędzia. Żadną filozofią nie jest ślepe podążanie za trendami i próbowanie implementacji popularnych silników bazodanowych do systemów, które wcale nie potrzebują grafowego lub kolumnowego podejścia.

HDD?

Bazy NoSQL-owe kończą przeżywać swój złoty wiek. Zaczynają być wykorzystywane w miejscach, w których powinny być wykorzystywane, a nie praktycznie wszędzie bez żadnego uzasadnienia.

W branży istnieje żartobliwe określenie HDD - *hype driven development* - które oznacza wykorzystywanie technologii, które są akurat *na fali*. Przestrzegam i nie polecam takiego podejścia w warunkach komercyjnych. A w prywatnych - czemu nie?

Bibliografia i ciekawe źródła



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



http://www.javaexpress.pl/article/show/DB40__Object_Database



<https://www.mongodb.com/compare/mongodb-mysql>



<https://blog.daftcode.pl/hype-driven-development-3469fc2e9b22>

Pytania?

Kod prezentacji dostępny jest w repozytorium git pod adresem
<https://bitbucket.org/krewak/pwsz-ppsi>



Wszystkie informacje dot. kursu dostępne są pod adresem
<http://pwsz.rewak.pl/kursy/4>

