

Baze de date Introducere

Nicolae-Cosmin Vârlan

October 1, 2020

Prima bază de date:





Cal (15.000 î.H. - 13.000 î.H.) - Lascaux, Franța



Câte animale am (7.000 î.H.) - Rio Pinturas, Argentina



Bazorelief (3200 î.H. - 400) - Egipt



Scrierea cuneiformă (600 î.H. - 300 î.H.) - Persia



Mai recent. . .



Stocarea datelor: foarte interesant este faptul că datele sunt “citite” în paralel - stocarea melodiilor pare foarte tentantă.





Idea de cilindru a fost păstrată, dar datele au fost scrise mult mai dens. Se observă și citirea “serială”



Citirea “serială”, de această dată pe un disc.



Discurile pe care le folosim azi au o densitate foarte mare și pot fi “citite” cu aprox. 200Mb/s (atenție la unele site-uri care zic că:)

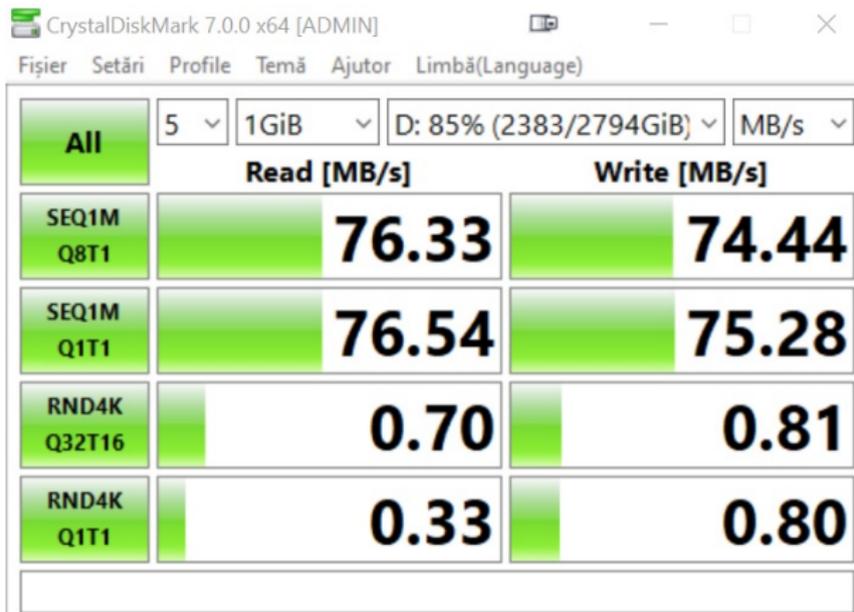
Specificatii

CARACTERISTICI GENERALE

| | |
|-----------------------|----------|
| Capacitate | 2 TB |
| Viteza de rotatie | 7200 rpm |
| Buffer | 64 MB |
| Interfata | SATA III |
| Rata de transfer SATA | 600 MB/s |
| Format | 3.5 inch |

De fapt ei nu zic nimic greșit, portul SATA III chiar merge cu 600MB/s, dar nu și HDD-urile clasice (cu platane).
[imagine preluată de pe situl eMAG]

Realitatea este puțin diferită (aceasta este viteza HDD-ului meu de acasă):





Este adevarat că SSD-urile ajung la viteze mai mari... DAR...  15/ 57

Evident, de această dată se prezintă (și) realitatea despre SSD:

Specificatii

CARACTERISTICI GENERALE

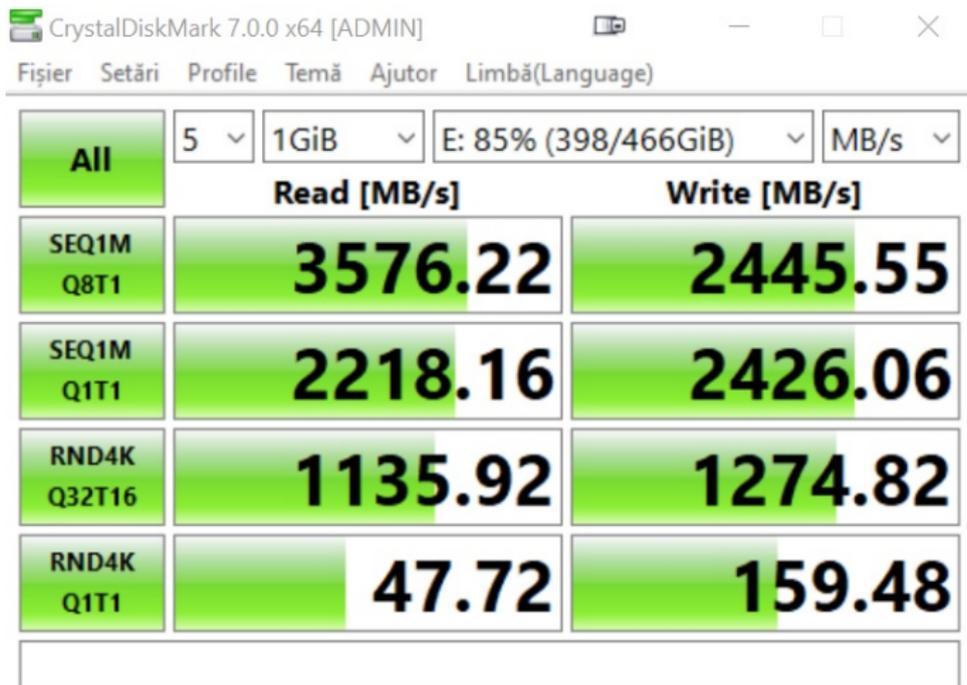
| | |
|------------------------------------|--------|
| Form Factor | 2.5" |
| Capacitate | 500 GB |
| Interfata | SATA 3 |
| Rata de transfer la citire (MB/s) | 550 |
| Rata de transfer la scriere (MB/s) | 520 |
| Rata de transfer SATA (MB/s) | 600 |

Se pare că viteza SATA III este prezentată ca o limitare (ceva de genul "de asta nu se poate mai mult")

Cu toate acestea, există SSDuri pe interfețe mai “noi” decât SATA. Este cazul SSD-urilor M.2 care folosesc standardul NVMe (Non Volatile Memory Express)



Primii care au reușit să atingă viteze de 3600MB/s la citire sunt cei de la Samsung.



Ați făcut aplicații care să utilizeze o bază de date ? [ce limbaj ?]



De ce credeți că sunt importante ?

De ce nu aș scrie într-un fișier informațiile ?



De ce nu aș scrie într-un fișier informațiile ?

Iată câteva posibile răspunsuri:

- ▶ separarea și izolarea datelor (no joins);
- ▶ posibilitatea duplicării datelor în mai multe fișiere (data integrity ?);
- ▶ inter-dependența datelor (chei primare / străine);
- ▶ formatele incompatibile ale diferitelor aplicații (logica stocării datelor este hardcodată în aplicație și datele sunt dependente de aplicație);
- ▶ interogări fixe specifice fiecărui fișier (nu există un limbaj de obținere a datelor);

- ▶ un fișier este stocat pe un singur calculator, nu au acces mai mulți utilizatori; chiar dacă ar fi distribuit (e.g. Samba) tot nu poate fi editat simultan de mai mulți utilizatori.
- ▶ ACID trebuie construit la nivel manual de fiecare dată pentru a permite accesul mai multor utilizatori (simultan) la informație;

Dezavantaje DBMS ?

- ▶ fișiere de dimensiuni mari;
- ▶ aplicații complexe ce depind de mentenanța exterioară;
- ▶ costuri (uneori mari);
- ▶ costuri pentru hardware;
- ▶ fișierele s-ar putea să fie accesate mai rapid dacă știi ce vreau;
- ▶ toți se bazează pe același DBMS - dacă acesta este închis. . .

Sisteme de gestiune a bazelor de date (SGBD)

Un sistem de gestiune de baze de date este alcătuit din:

- ▶ Hardware
- ▶ Software
- ▶ Utilizatori
- ▶ Date

Scopul său este de a deservi **rapid** foarte **mulți utilizatori** care fac interogări diferite într-o **cantitate foarte mare de date**.

Securitate * Acces controlat la baza de date * Stocarea, regăsirea, actualizarea datelor * Integritate * Suport pentru tranzații * Control concurrent * Recuperarea datelor * Catalog (dicționar de date)

- ▶ Hardware
- ▶ Software
- ▶ Utilizatori
- ▶ Date

Hardware-ul poate varia de la un simplu PC până la rețele de calculatoare și trebuie să asigure:

- ▶ Persistența datelor (chiar în cazuri critice).
- ▶ Stocarea unui volum mare de date.
- ▶ Accesul rapid la date (vezi discuția despre HDD-uri).

- ▶ Hardware
- ▶ Software
- ▶ Utilizatori
- ▶ Date

Hardware-ul poate varia de la un simplu PC până la rețele de calculatoare și trebuie să asigure:

- ▶ Persistența datelor (chiar în cazuri critice).
- ▶ Stocarea unui volum mare de date.
- ▶ Accesul rapid la date (vezi discuția despre HDD-uri).

- ▶ Hardware
- ▶ Software
- ▶ Utilizatori
- ▶ Date

Partea de software trebuie să ofere (măcar) o metodă de definire a datelor și una de manipulare a acestora [fox]:

- ▶ DDL (Data Definition Language)
- ▶ DML (Data Manipulation Language)

Dar are și rolul de management a utilizatorilor (Data Control Language - *grant, revoke*), realiza conexiuni cu software extern ce dorește să acceseze informațiile din BD, control al tranzacțiilor (Transaction Control Language - *savepoint, commit, rollback*), etc.

- ▶ Hardware
- ▶ Software
- ▶ Utilizatori
- ▶ Date

Persoanele care interacționează cu baza de date:

- ▶ Administratorul
- ▶ Proiectantul
- ▶ Programatorii de aplicații
- ▶ Utilizatorii ce folosesc aplicațiile

- ▶ Hardware
- ▶ Software
- ▶ Utilizatori
- ▶ **Date**

Datele vor fi stocate în fișiere având formate specifice.

De-a lungul timpului au existat mai multe modele de baze de date. Modelul relațional, pe care îl studiem, este în prezent cel utilizat de SGBD-uri precum MySQL, MariaDB, Oracle, Postgresql, SQL Server, SQL Lite, etc.

Același limbaj: SQL.

Modele de baze de date

În trecut:

- ▶ Modelul ierarhic (IBM's IMS, sf. '60)
- ▶ Modelul rețea (CODASYL 1971)
- ▶ Modelul relațional (Codd, '70)
- ▶ Modelul obiect-relațional ('90)

În prezent:

- ▶ Modelul relațional (cel al lui Codd)
- ▶ Diverse modele nerelaționale (2000s) - peste 225 variante (despre care puteți citi la <http://nosql-database.org/>) ce nu constituie subiectul acestui curs.

Modele de baze de date

În trecut:

- ▶ Modelul ierarhic (IBM's IMS, sf. '60)
- ▶ Modelul rețea (CODASYL 1971)
- ▶ Modelul relațional (Codd, '70)
- ▶ Modelul obiect-relațional ('90)

În prezent:

- ▶ Modelul relațional (cel al lui Codd)
- ▶ Diverse modele nerelaționale (2000s) - <http://nosql-database.org/>

Modele de baze de date - modelul ierarhic

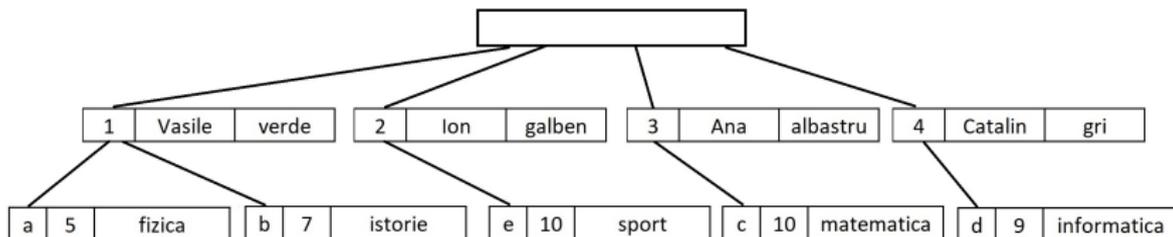
Datele sunt organizate într-un arbore, fiecare nod reprezentând o linie din tabel. Baza de date are și un nod rădăcina care nu conține nimic.

Exemplu:

$S = \{1\text{-Vasile-verde}, 2\text{-Ion-galben}, 3\text{-Ana-albastru}, 4\text{-Catalin-gri}\}$

$N = \{a\text{-5-fizica}, b\text{-7-istorie}, c\text{-10-matematica}, d\text{-9-informatica}, e\text{-10-sport}\}$

$Muchii = \{(\text{null},1), (\text{null},2), (\text{null},3), (\text{null},4), (1,a), (1,b), (2,e), (3,c), (4,d)\}$



Modelul ierarhic permite unui nod sa aibă doar un singur părinte.

Modele de baze de date

În trecut:

- ▶ Modelul ierarhic (IBM's IMS, sf. '60)
- ▶ Modelul rețea (CODASYL 1971)
- ▶ Modelul relațional (Codd, '70)
- ▶ Modelul obiect-relațional ('90)

În prezent:

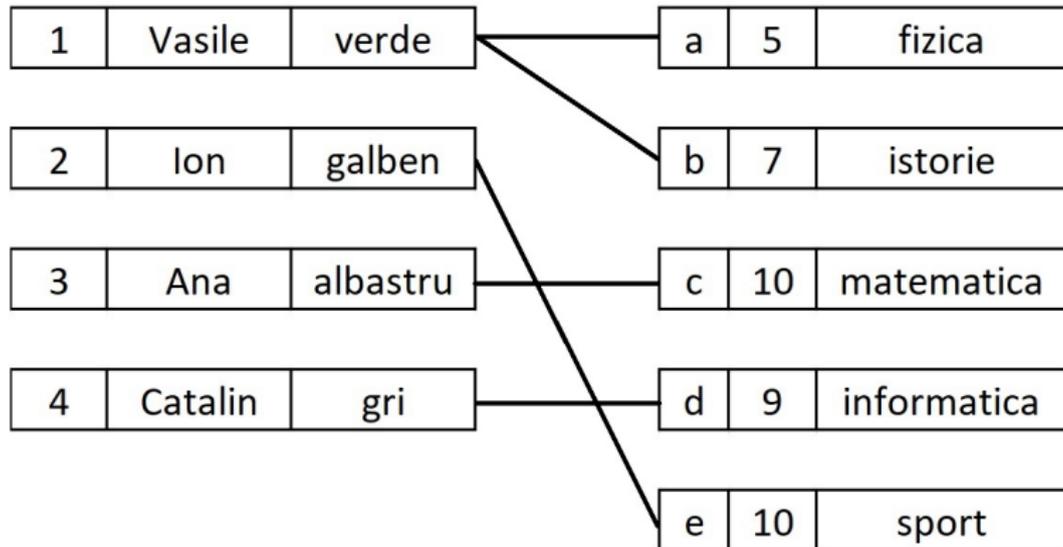
- ▶ Modelul relațional (cel al lui Codd)
- ▶ Diverse modele nerelaționale (2000s) - <http://nosql-database.org/>

Modele de baze de date - modelul rețea

Considerăm o mulțime de studenți și o mulțime de note. În modelul ierarhic se construiește o a treia mulțime de cupluri din conținând identificatori ai elementelor primei mulțimi respectivi ai elementelor celei de-a doua mulțimi.

Exemplu:

$$S = \{1\text{-Vasile-verde}, 2\text{-Ion-galben}, 3\text{-Ana-albastru}, 4\text{-Catalin-gri}\}$$
$$N = \{a\text{-5-fizica}, b\text{-7-istorie}, c\text{-10-matematica}, d\text{-9-informatica}, e\text{-10-sport}\}$$
$$C = \{(1,a), (1,b), (2,e), (3,c), (4,d)\}$$



Modele de baze de date

În trecut:

- ▶ Modelul ierarhic (IBM's IMS, sf. '60)
- ▶ Modelul rețea (CODASYL 1971)
- ▶ Modelul relațional (Codd, '70)
- ▶ Modelul obiect-relațional ('90)

În prezent:

- ▶ Modelul relațional (cel al lui Codd)
- ▶ Diverse modele nerelaționale (2000s) - <http://nosql-database.org/>

Modele de baze de date - modelul relațional

În modelul relațional se stabilesc care sunt entitățile ce trebuie să fie memorate în baza de date și se construiesc asocieri între tabele. În exemplul nostru, entitățile sunt elevii și notele sunt doar asociate acestora. În continuare informațiile sunt stocate ca mulțimi, relațiile dintre elementele acestora este realizată în baza unor chei primare / străine.

Exemplu:

$S = \{1\text{-Vasile-verde}, 2\text{-Ion-galben}, 3\text{-Ana-albastru}, 4\text{-Catalin-gri}\}$

$N = \{a\text{-5-fizica-1}, b\text{-7-istorie-1}, c\text{-10-matematika-3}, d\text{-9-informatica-4}, e\text{-10-sport-2}\}$

| id | nume | culoare |
|----|---------|----------|
| 1 | Vasile | verde |
| 2 | Ion | galben |
| 3 | Ana | albastru |
| 4 | Catalin | gri |

| id | nota | materie | ref_id |
|----|------|-------------|--------|
| a | 5 | fizica | 1 |
| b | 7 | istorie | 1 |
| c | 10 | matematica | 3 |
| d | 9 | informatica | 4 |
| e | 10 | sport | 2 |

Informația din coloana "id" din primul tabel este cea pe baza careia se face legătura dintre cele două tabele. Valorile din coloana "ref_id" din cel de-al doilea tabel fac referință către această colană.

Modele de baze de date

În trecut:

- ▶ Modelul ierarhic (IBM's IMS, sf. '60)
- ▶ Modelul rețea (CODASYL 1971)
- ▶ Modelul relațional (Codd, '70)
- ▶ **Modelul obiect-relațional ('90)** - impedance mismatch problem

În prezent:

- ▶ Modelul relațional (cel al lui Codd)
- ▶ Diverse modele nerelaționale (2000s) - <http://nosql-database.org/>

În prezent:

- ▶ Modelul relațional (cel al lui Codd)
- ▶ Diverse modele nerelaționale (2000s) - <http://nosql-database.org/>

Relațional vs Nerelațional - Oare cine câștigă ?

În prezent:

- ▶ Modelul relațional (cel al lui Codd)
- ▶ Diverse modele nerelaționale (2000s) - <http://nosql-database.org/>

Relațional vs Nerelațional - Oare cine câștigă ?

Dar mai întâi, o poveste: cea a dublei rezervări a unei camere de hotel...

În prezent:

- ▶ Modelul relațional (cel al lui Codd)
- ▶ Diverse modele nerelaționale (2000s) - <http://nosql-database.org/>

Relațional vs Nerelațional - Oare cine câștigă ?

Proprietățile ce s-ar dori pentru un sistem (distribuit):

- ▶ Consistență (C)onsistency
- ▶ Disponibilitate (A)vailability
- ▶ Distribuirea pe mai multe calculatoare (P)artition Tolerance

Teorema CAP afirmă că un sistem distribuit poate satisface doar două dintre aceste proprietăți.

CAP (continuare / demonstrație)

Sistemele distribuite (presupunem formate din două calculatoare C_1 și C_2) se bazează pe rețea care poate eșua (fail). Atunci:

A) Păstrarea disponibilității: C_1 poate continua operațiile fără a se sincroniza cu C_2 și poate obține rezultate care din cauză că depindeau de C_2 sunt eronate (pentru că nu a putut obține informații de la C_2) - de exemplu poate vinde stocul de produse ce a fost deja vândut de C_2 (dar va fi disponibil).

B) Păstrarea consistenței: C_1 poate să NU servească nici un client pentru a nu obține date eronate (dar va fi consistent).

Evident, dacă rețeaua funcționează, C_1 poate să se sincronizeze cu C_2 și sincronizarea va duce atât la disponibilitate cât și la consistență. Pot să fiu sigur că rețeaua funcționează doar dacă nu depind de ea; deci sistemul nu mai e distribuit / partiționat.

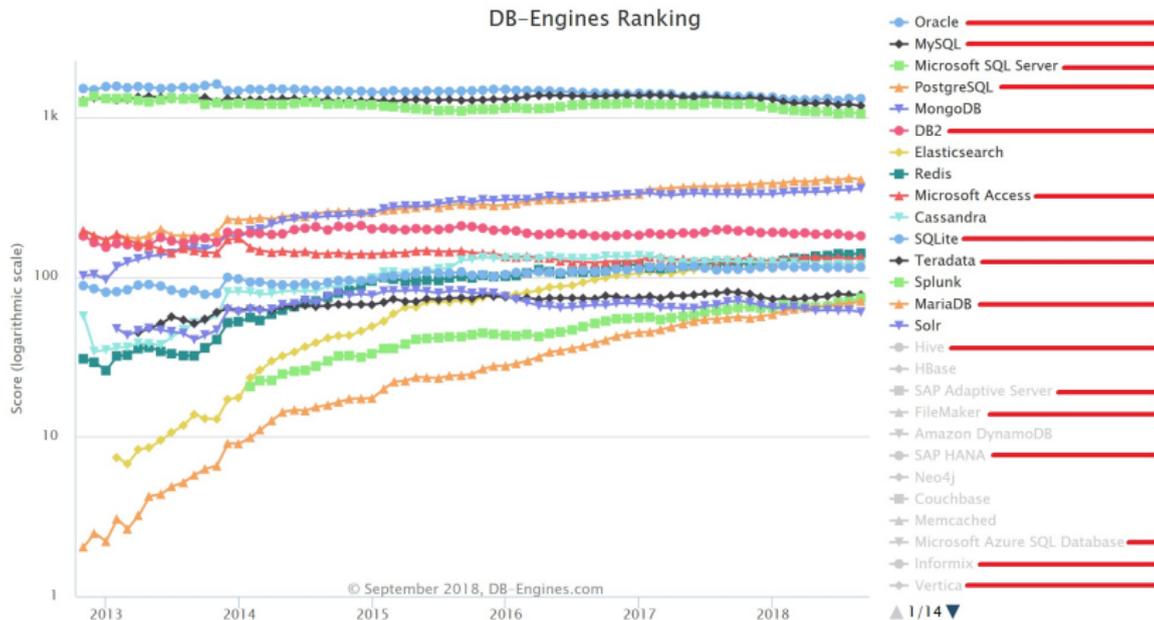
Tranzacția în BD relaționale este ACID

- ▶ **Atomică** - Toate operațiile din tranzacție sunt executate (ca o singură entitate atomică) sau, în caz contrar, toate sunt ignorate și BD revine la starea inițială.
- ▶ **Consistență** - La sfârșitul tranzacției, datele sunt așa cum ar trebui să fie conform logicii tranzacției.
- ▶ **Izolarea** - Tranzacțiile nu știu una de alta și nu se influențează, chiar dacă sunt executate simultan. Fiecare tranzacție crede că în momentul în care ea se execută, nimeni altcineva nu mai interacționează cu BD.
- ▶ **Durabilitate** - Dacă tranzacția s-a terminat și s-a făcut commit, nimeni și nimic nu o mai pot aduce într-o stare anterioară.

Tranzacția în BD nerelaționale este BASE

- ▶ **Basic Availability** - Baza de date pare că este funcțională de cele mai multe ori (adică, "în principiu este disponibilă").
- ▶ **Soft-state** - Locurile din rețea unde este stocată baza de date nu trebuie neapărat să fie consistente unele cu celelalte (deși se încearcă).
- ▶ **Eventual consistency** - Uneori datele vor fi consistente (când se sincronizează)

Relațional vs Nerelațional; De ce Oracle ?

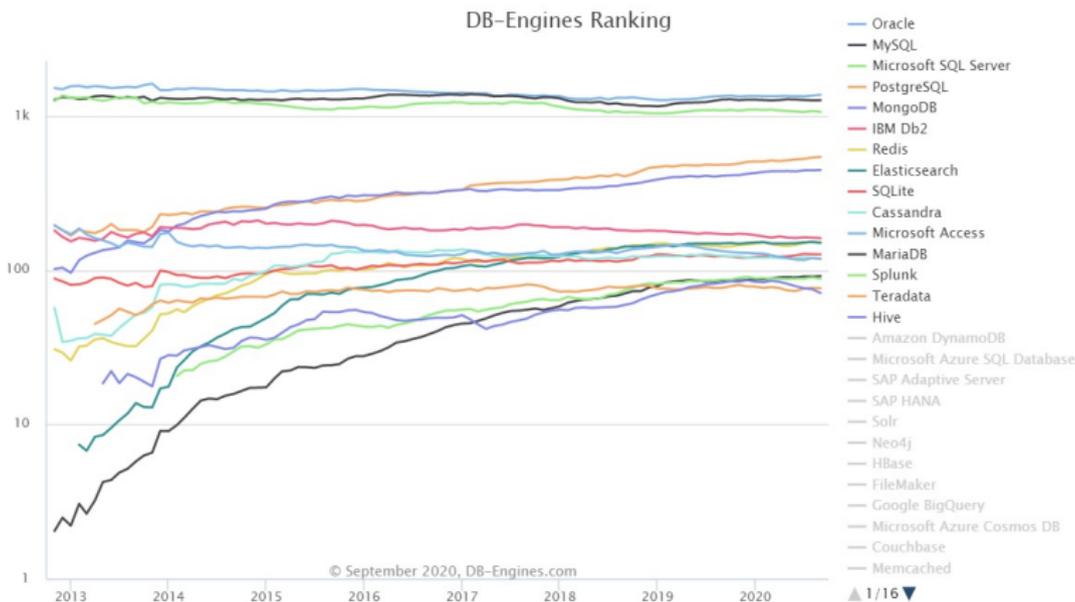


<https://db-engines.com> (01-oct-2018)



46 / 57

Relațional vs Nerelațional; De ce Oracle ?



<https://db-engines.com> (01-oct.-2020)

| Rank | | | DBMS | Database Model | Score | | |
|----------|----------|----------|------------------------------|----------------------------|----------|----------|----------|
| Sep 2020 | Aug 2020 | Sep 2019 | | | Sep 2020 | Aug 2020 | Sep 2019 |
| 1. | 1. | 1. | Oracle + | Relational, Multi-model | 1369.36 | +14.21 | +22.71 |
| 2. | 2. | 2. | MySQL + | Relational, Multi-model | 1264.25 | +2.67 | -14.83 |
| 3. | 3. | 3. | Microsoft SQL Server + | Relational, Multi-model | 1062.76 | -13.12 | -22.30 |
| 4. | 4. | 4. | PostgreSQL + | Relational, Multi-model | 542.29 | +5.52 | +60.04 |
| 5. | 5. | 5. | MongoDB + | Document, Multi-model | 446.48 | +2.92 | +36.42 |
| 6. | 6. | 6. | IBM Db2 + | Relational, Multi-model | 161.24 | -1.21 | -10.32 |
| 7. | 7. | ↑ 8. | Redis + | Key-value, Multi-model | 151.86 | -1.02 | +9.95 |
| 8. | 8. | ↓ 7. | Elasticsearch + | Search engine, Multi-model | 150.50 | -1.82 | +1.23 |
| 9. | 9. | ↑ 11. | SQLite + | Relational | 126.68 | -0.14 | +3.31 |
| 10. | ↑ 11. | 10. | Cassandra + | Wide column | 119.18 | -0.66 | -4.22 |
| 11. | ↓ 10. | ↓ 9. | Microsoft Access | Relational | 118.45 | -1.41 | -14.26 |
| 12. | 12. | ↑ 13. | MariaDB + | Relational, Multi-model | 91.61 | +0.69 | +5.54 |
| 13. | 13. | ↓ 12. | Splunk | Search engine | 87.90 | -2.01 | +0.89 |
| 14. | 14. | ↑ 15. | Teradata + | Relational, Multi-model | 76.39 | -0.39 | -0.57 |
| 15. | 15. | ↓ 14. | Hive | Relational | 71.17 | -4.12 | -11.93 |
| 16. | 16. | ↑ 18. | Amazon DynamoDB + | Multi-model | 66.18 | +1.43 | +8.36 |
| 17. | 17. | ↑ 25. | Microsoft Azure SQL Database | Relational, Multi-model | 60.45 | +3.60 | +32.91 |
| 18. | 18. | ↑ 19. | SAP Adaptive Server | Relational | 54.01 | +0.05 | -2.09 |
| 19. | 19. | ↑ 21. | SAP HANA + | Relational, Multi-model | 52.86 | -0.26 | -2.53 |
| 20. | 20. | ↓ 16. | Solr | Search engine | 51.62 | -0.08 | -7.35 |

<https://db-engines.com> (01-oct.-2020)

Concluzia ?

Bazele de date relaționale și cele nerelaționale vor coexista, fiecare având avantajele lor.

Cu siguranță cele relaționale nu vor dispărea în următoarele două secole.

Dovada:

Enterprise still uses SQL in 23th century [Star Trek Discovery]

Modelul relațional - concepte (intuitiv)

Momentan vom da câteva concepte la nivel intuitiv, dar vom formaliza în cursul viitor.

| id | nume | culoare |
|----|---------|----------|
| 1 | Vasile | verde |
| 2 | Ion | galben |
| 3 | Ana | albastru |
| 4 | Catalin | gri |

Relație = Tabel (cu roșu)

Modelul relațional - concepte (intuitiv)

| id | nume | culoare |
|----|---------|----------|
| 1 | Vasile | verde |
| 2 | Ion | galben |
| 3 | Ana | albastru |
| 4 | Catalin | gri |

Atribute (cu roșu)

Modelul relațional - concepte (intuitiv)

| id | nume | culoare |
|----|---------|----------|
| 1 | Vasile | verde |
| 2 | Ion | galben |
| 3 | Ana | albastru |
| 4 | Catalin | gri |

Tuplu sau linie

Modelul relațional - concepte (intuitiv)

| id | nume | culoare |
|----|---------|----------|
| 1 | Vasile | verde |
| 2 | Ion | galben |
| 3 | Ana | albastru |
| 4 | Catalin | gri |

Valoare (a atributului nume din tuplul t_3)

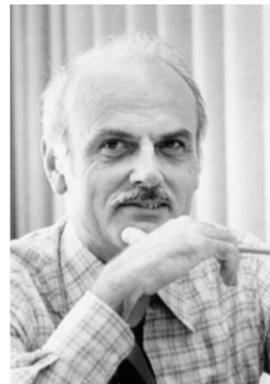
Modelul relațional - concepte (intuitiv)

| id | nume | culoare |
|----|---------|----------|
| 1 | Vasile | verde |
| 2 | Ion | galben |
| 3 | Ana | albastru |
| 4 | Catalin | gri |

Domeniul unui atribut = mulțimea de elemente din care se poate selecta valoarea unui atribut (e.g. {Vasile, Ion, Ana, Catalin})

Modelul relațional

- ▶ Fiecare element al relației conține informație (null este informație).
- ▶ Fiecare atribut este unic.
- ▶ Valorile unui atribut sunt din același domeniu.
- ▶ Nu există linii identice în tabel.
- ▶ Ordinea rândurilor și coloanelor este arbitrară.
- ▶ Are la bază algebra relațională introdusă de Edgar Frank Codd.



E.F. Codd

Modelul relațional - chei

- ▶ **Supercheie** - un atribut sau o mulțime de atribute care identifică unic un tuplu într-o relație
- ▶ **Cheie candidat** - o supercheie cu proprietatea că nici o submulțime proprie a sa nu este supercheie
- ▶ **Cheie primară** - o cheie candidat selectată pentru a identifica în mod unic tuplele într-o relație
- ▶ **Cheie alternativă** - Chei candidat care nu au fost selectate pentru a juca rolul de cheie primară
- ▶ **Cheie străină** - un atribut sau o submulțime de atribute dintr-o relație care face referință la o cheie candidat a altei relații

Bibliografie

- ▶ Database Systems - A Practical Approach to Design, Implementation, and Management, *Thomas Connolly, Carolyn Begg*; Pearson, 2015
- ▶ Database System Concepts, *Abraham Silberschatz, Henry F. Korth, S. Sudarshan*; McGrawHill, 2011