Modele de date relationale versus non relationale - Aplicatii

Ioana Ciuciu
ioana.ciuciu@ubbcluj.ro
http://www.cs.ubbcluj.ro/~oana/

Planificare

Sunt posibile mici modificari ale planificarii in timpul semestrului

Saptamana	Curs	Seminar	Laborator
S1	1. Concepte fundamentale ale bazelor de date. Modelare conceptuala	1. Modelul Entitate-Relatie. Modelul relational	1. Modelarea unei BD in modelul ER si implementarea ei in SQL Server
S2	2. Modelul relational de organizare a bazelor de date. Modelare conceptuala		
S3	3. Gestiunea bazelor de date relationale cu limbajul SQL (DDL)	2. Limbajul SQL – definirea si actualizarea datelor	2. Interogari SQL
S4	4. Gestiunea bazelor de date relationale cu limbajul SQL (DML)		
S5-6	5-6. Dependente functionale, forme normale	3. Limbajul SQL – regasirea datelor	3. Interogari SQL avansate
S7	7. Interogarea bazelor de date relationale cu operatori din algebra relationala	4. Proceduri stocate	4. Proceduri stocate. View. Trigger
S8	8. Structura fizica a bazelor de date relationale		
S9	9. FARA CURS (30 nov.)	5. View-uri. Functii definite de utilizator. Trigger	
S10-11	10-11. Indecsi. Arbori B. Fisiere cu acces direct	6. Formele normale ale unei relatii. Indecsi	
S12	12. Extensii ale modelului relational si baze de date NoSQL – curs invitat UBB, vineri 22 Dec. 2023		
S13	13. Evaluarea interogarilor in bazele de date relationale	7. Probleme	Examen practic
S14	14. Aplicatii		

Planul cursului

- Curs 14
  - Modelul relational
  - Modelul non relational
  - Analiza comparativa

Modelul Relational

- **Modelul relational** a fost propus de catre Edgar F. Codd in anul1970
  - Concepte introduse: independenta datelor, format tabelar, limbaj de interogare a datelor SQL (Structured Query Language)
- Orice operatie sau grup de operatii (tranzactie) asupra unei bazei de date relationale trebuie sa respecte **modelul ACID**:
  - **Atomicitate**: tranzactia este tratata ca o singura unitate (daca o singura operatie nu se poate executa, atunci se anuleaza efectul intregii tranzactii)
  - **Consistenta**: baza de date este consistenta la sfarsitul executiei tranzactiei (respecta regulile de integritate memorate)
  - **Izolare**: pe timpul executiei unei tranzactii T, alte tranzactii nu pot modifica datele gestionate de tranzactia T
  - **Durabilitate**: daca o tranzactie s-a terminat de executat, atunci sistemul asigura ca ea nu mai trebuie re-executata in cazul unor erori
  - Exista foarte multe aplicatii unde un astfel de sistem tranzactional este foarte important, de exemplu in *domeniul bancar*

Modelul Relational

- Bazele de date relationale sunt instrumente traditionale robuste pentru partea de backend a aplicatiilor online (magazinelor/comert online)
- Acestea sunt folosite in multe aplicatii pentru cosuri de cumparaturi, cataloage produs, si alte asemenea date
- In acest caz, de ce nu sunt potrivite pentru a indeplini cerintele unui gigant ca, de exemplu, Amazon?

Modelul Relational

- Desi foarte folosit la ora actuala in aplicatii apartinand unui spectru larg de domenii (comert online, productie, bancar, etc.), modelul de date relational are unele limitari tehnice:
  - Baza de date este formata dintr-o multime de relatii (tabele). Fiecare relatie are o **schema fixa**. Modificarea acestei scheme (deci structura virtuala a bazei de date) este greoaie si poate sa implice costuri mari
  - Este dificil de a interoga date care depind de alte date (din alte baze de date), deci este necesara o **interconectare a datelor**
  - Dimensiunea tabelelor poate creste foarte mult (de exemplu, in bazele de date unde este necesar sa fie memorate cantitati foarte mari de text si informatii binare: imagini, audio, film)
  - Pentru a asigura **scalabilitatea**, este necesara distribuirea bazei de date pe mai multe servere, ceea ce face foarte dificila/imposibila gestiunea si mentenanta tabelelor

Nevoia unei alternative non-traditionala (non-relationala) de reprezentare a datelor

- Doua exemple de studii de caz reale care au adoptat arhitecturi non-traditionale inca de timpuriu:
  - last.fm
  - Amazon

Nevoia unei alternative non-traditionala (non-relationala) de reprezentare a datelor

- **Last.fm** - radio communitate online in jurul muzicii, furnizeaza:
  - Streamuri de muzica & download-uri
  - **Servicii personalizate**, ca de exemplu recomandari de muzica si de evenimente
  - Liste personalizate
  - Prezentare conforma cu obiceiurile personale de ascultare, gusturi in muzica, muzica similara
  - In particular, furnizeaza aceste servicii pentru **foarte multi utilizatori in paralel**
- In 2012, avea mai mult de 25 milioane de utilizatori / luna

Nevoia unei alternative non-traditionala (non-relationala) de reprezentare a datelor

- **Last.fm** - *internet radio* si *music community website*, furnizeaza
  - Streamuri de muzica & download-uri
  - **Servicii personalizate**, ca de exemplu recomandari de muzica si de evenimente
  - Liste personalizate
  - Prezentare conforma cu obiceiurile personale de ascultare, gusturi in muzica, muzica similara si similaritati de
  - In particular, furnizeaza aceste servicii pentru **foarte multi utilizatori in paralel**
- In 2012, avea mai mult de 25 milioane de utilizatori / luna
- Aceasta pune in evidenta aspectul *stocarii distribuite redundante a unor mari cantitati de date*!

Nevoia unei alternative non-traditionala (non-relationala) de reprezentare a datelor

- **Amazon** – companie e-commerce in US
  - Gestioneaza multe aplicatii separate care folosesc volume mari de date, ca de exemplu
    - Listele vanzatorilor
    - Cosuri de cumparaturi
    - Preferintele clientilor
    - Management de sesiune
    - Ranking-ul vanzarilor
    - Cataloage produs
- Website-urile sunt redate folosind date, agregari si analiza rezultatelor oferite de diferite aplicatii

Nevoia unei alternative non-traditionala (non-relationala) de reprezentare a datelor

- Aceasta conduce la urmatoarele cerinte ale infrastructurii Amazon:
  - **Disponibilitate ridicata** - pentru operatii citire-scriere
  - **Performanta ridicata** - timpi scazuti de raspuns
  - **Service-level agreement** intre applicatii - timpi de raspuns scazuti garantati chiar si pentru sarcina (computationala) ridicata

Nevoia unei alternative non-traditionala (non-relationala) de reprezentare a datelor

- Aceasta conduce la urmatoarele cerinte ale infrastructurii Amazon:
  - **Disponibilitate ridicata** - pentru operatii citire-scriere
  - **Performanta ridicata** - timpi scazuti de raspuns
  - **Service-level agreement** intre applicatii - timpi de raspuns scazuti garantati chiar si pentru sarcina (computationala) ridicata
- Aceasta pune in evidenta problema bazelor de date a caror dimensiune depaseste capacitatea bazelor de date traditionale!

Nevoia unei alternative non-traditionala (non-relationala) de reprezentare a datelor

- In mod cronologic, last.fm si Amazon ofera exemple a doua aspecte diferite :
  - (1) stocare distribuita redundanta a unor cantitati mari de date si
  - (2) baze de date a caror dimensiune depaseste capacitatea bazelor de date traditionale
- Aceste aspect conduc la doua modele de sisteme de date de volum mare (big data):
  - Modelul 1: Hadoop
  - Modelul 2: Amazon Dynamo

Modelul 1: Hadoop

- Last.fm foloseste Apache Hadoop incepand cu anul 2006 ca infrastructura pentru a face fata costului computational ridicat determinat de baza sa extinsa de utilizatori
- Ce anume face ca Hadoop sa fie o solutie buna in acest caz?
  - Hadoop furnizeaza un system de fisiere distribuit cu backup-uri redundante
  - Face posibila scalabilitatea cu "commodity hardware"
  - E disponibil gratuit
  - E un system open source care poate fi extins
  - Flexibil si usor de invatat
- In 2010 last.fm gestiona in jur de 100 TB de date folosind 50 noduri cu un total de 300 procesoare

Modelul 1: Hadoop

- Facebook detine unul dintre cele mai mari clustere Hadoop din intreaga lume (deja cu cativa ani inainte, clusterul FB avea 9 TB de memorie centrala) pentru a oferi utilizatorilor sai:
  - Rapoarte zilnice/la fiecare ora
    - Pentru analiza si planificare de produs
    - Performanta campaniilor de marketing
    - Rapoarte ale datelor relative la utilizare (usage data)
  - Evaluari ad hoc de date istorice
  - Arhivare pe termen lung a datelor

Model 2: Amazon Dynamo

- Amazon se confrunta cu volume de date extrem de mari
- Chiar daca datele Amazon sunt structurate astfel incat sa sugereze folosirea sistemelor de baze de date traditionale, cantitatea enorma a datelor face imposibila folosirea acestora
- Pentru acest motiv, dar nu numai, Amazon a dezvoltat propriul system de baze de date specific big data, numit Dynamo
- Dynamo vine insotit de urmatoarele principii de design:
  - **Scalabilitate incrementala**: Dynamo ar trebui sa fie capabil sa fie extins cu cate un nod o data
  - **Simetrie**: Fiecare nod in Dynamo ar trebui sa aibe aceleasi responsabilitati ca si nodurile vecine
  - **Decentralizare**: Tehnici decentralizate peer-to-peer
  - **Heterogenitate**: Sistemul trebuie sa poata folosi heterogenitatea infrastructurii pe care ruleaza. Acest lucru e esential pentru a adauga noi noduri cu capacitate mai crescuta fara a fi nevoie sa fie upgradate toate host-urile in acelasi timp.
- Acestea, impreuna cu Google BigTable au marcat inceputul unei noi generatii de sisteme de baze de date proiectate special pentru a putea gestiona probleme specific big data

1 https://cloud.google.com/bigtable/

Nevoia unei alternative non-traditionala (non-relationala) de reprezentare a datelor

- Pentru rezolvarea acestor probleme - *stocarea, gestiunea și prelucrarea unor volume foarte mari de date* - s-au propus solutii **NoSQL** (**not only SQL**)
- Termenul a fost utilizat prima data in 1998 relativ la o baza de date care nu folosea SQL pentru gestiune
- Acest termen a fost reluat, cu semnificatia de azi, in anul 2009

Modelul non relational

- **Caracteristicile generale** ale solutiilor NoSQL:
  - memorarea si facilitarea analizei unor volume mari de date (companiile amintite mai sus foloseau intre 10-100K servere)
  - nu exista o structura fixa a datelor, **schema flexibila**
  - sunt proiectate pentru a fi **scalabile pe orizontala si pentru a functiona intr-un mediu distribuit**
  - intre date se pot stabili legaturi (prin referire la date memorate in alte baze de date)
  - aceleasi date pot sa fie memorate pe mai multe servere (**partajare si replicare**)
  - la interogare nu se folosesc operatii de join (mari consumatoare de timp)
  - sunt **solutii foarte bune pentru cazuri particulare, ad hoc** (NU pentru orice gestiune de date)
  - sunt open source

Modelul non relational

- Bazele de date NoSQL au devenit prima alternativa a bazelor de date relationale in primul rand datorita **scalabilitatii**, a **disponibilitatii datelor** si a **tolerantei la erori**

Modelul non relational

- **Dezavantaje** ale modelelor NoSQL:
  - nu exista standarde (cum exista standardul SQL la bazele de date relationale)
  - nu se asigura consistenta bazei de date (de catre sistemul de gestiune)
  - exista posibilitati limitate de interogare

Modelul non relational

- La sistemele **NoSQL**, modelul ACID este greu sa fie respectat (mai ales din cauza distribuirii si replicarii), si atunci el se inlocuieste cu **modelul BASE**:
  - **Basic Availability**: toti clientii primesc un raspuns la o interogare (in loc de a folosi o singura sursa de date, colectia de date este replicata si distribuita, deci undeva in retea datele cautate trebuie sa existe)
  - **Soft State**: consistenta bazei de date nu este verificata de SGBD, ea trebuie sa fie asigurata de clientul (programul) care are dreptul de modificare a bazei de date
  - **Eventual Consistency**: baza de date poate sa se afle intr-o stare de inconsistenta (exista valori diferite ale aceleasi date), dar se presupune ca in viitor datele vor ajunge intr-o stare de consistenta. Propagarea modificarilor la replicile datei va fi efectuata in viitor.

Modelul non relational

- In anul 2000 Eric Brewer a formulat o conjectura bazata pe experienta acestuia cu motorul de cautare Inktomi
- Aceasta conjectura (numita conjectura lui Brewer) a fost demonstrata doi ani mai tarziu, si denumita **Teorema CAP**:

Este imposibil pentru un serviciu web sa ofere simultan urmatoarele trei facilitati:

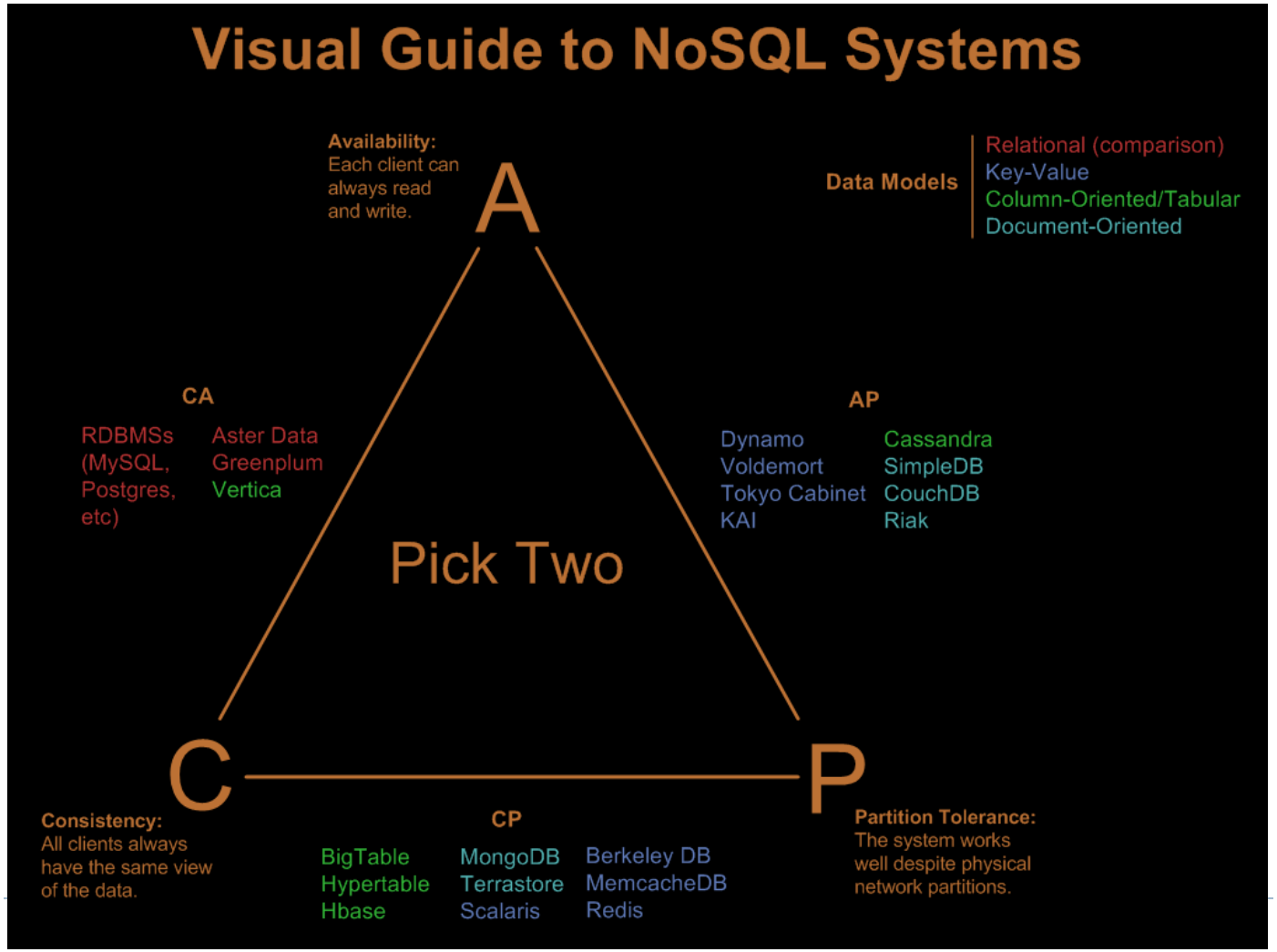
- Consistenta (**C**onsistency)
- Disponibilitate la cereri (**A**vailability)
- Partitionare (**P**artition)

Modelul non relational

- Folosind aceasta teorema, sistemele de baze de date se pot imparti in trei categorii (clase) dupa proprietatile pe care le au: **CA, AP, CP**.
- Deoarece sistemele NoSQL trebuie sa permita **partitionarea**, acestea sunt incluse in una din clasele:
  - **CP (Consistency + Partition)**: au un grad mai ridicat de consistena, in defavoarea disponibilitatii
  - **AP (Availability + Partition)**: au un grad mai ridicat de disponibilitate, in timp ce restrictiile cerute pentru consistenta s-au restrans (sau chiar eliminat)

Modelul non relational

In [Hurst] se da urmatoarea imagine cu privire la categoriile (clasele) amintite mai sus



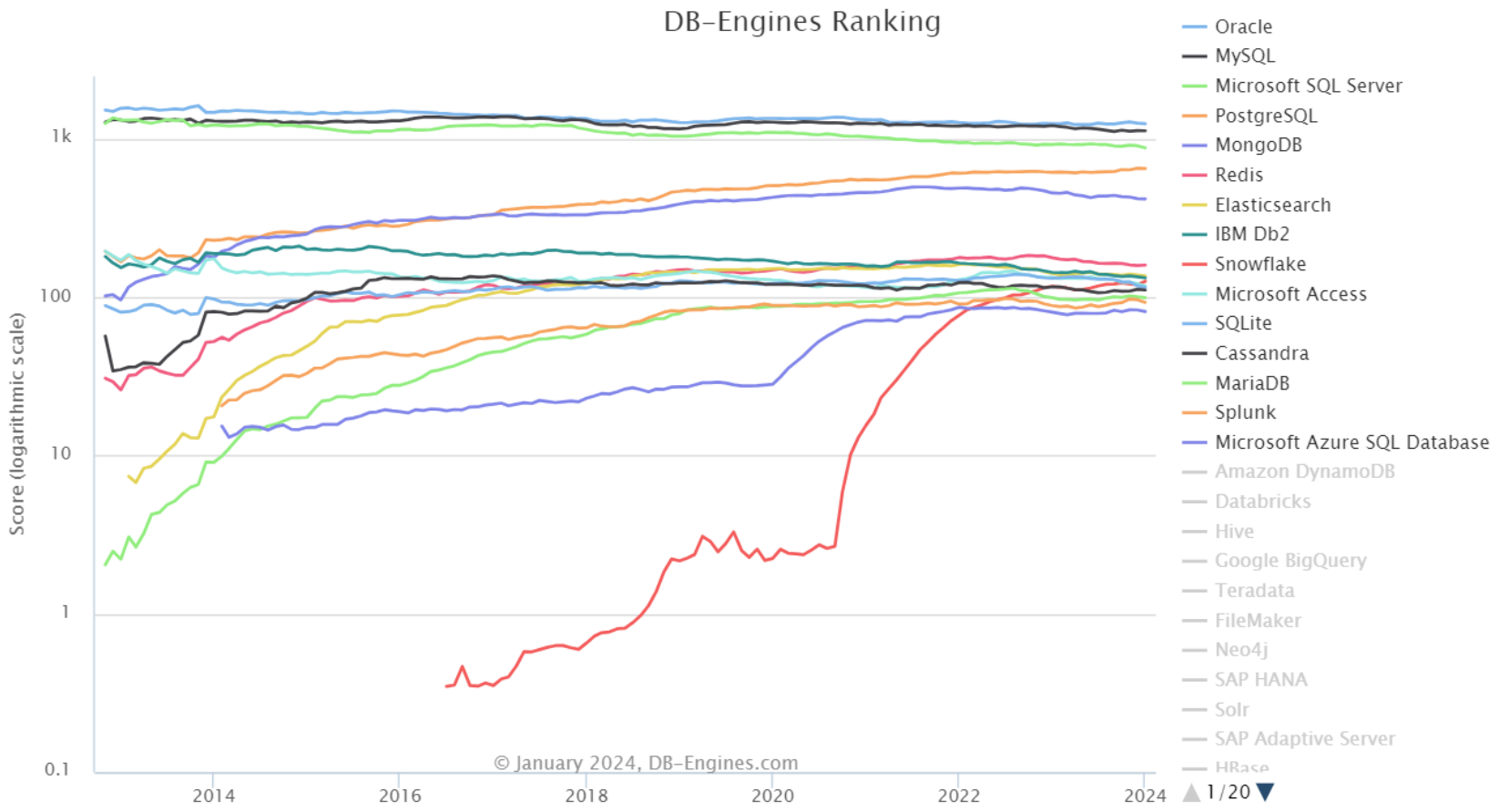
Modelul non relational

- Ranking-ul bazelor de date, dupa https://db-engines.com/en/ranking

417 systems in ranking, January 2024

Rank			DBMS	Database Model	Score		
Jan 2024	Dec 2023	Jan 2023			Jan 2024	Dec 2023	Jan 2023
1.	1.	1.	Oracle	Relational, Multi-model	1247.49	-9.92	+2.33
2.	2.	2.	MySQL	Relational, Multi-model	1123.46	-3.18	-88.50
3.	3.	3.	Microsoft SQL Server	Relational, Multi-model	876.60	-27.23	-42.79
4.	4.	4.	PostgreSQL	Relational, Multi-model	648.96	-1.94	+34.11
5.	5.	5.	MongoDB	Document, Multi-model	417.48	-1.67	-37.70
6.	6.	6.	Redis	Key-value, Multi-model	159.38	+1.03	-18.17
7.	7.	↑ 8.	Elasticsearch	Search engine, Multi-model	136.07	-1.68	-5.09
8.	8.	↓ 7.	IBM Db2	Relational, Multi-model	132.41	-2.19	-11.16
9.	↑ 10.	↑ 11.	Snowflake	Relational	125.92	+6.04	+8.66
10.	↓ 9.	↓ 9.	Microsoft Access	Relational	117.67	-4.08	-15.69

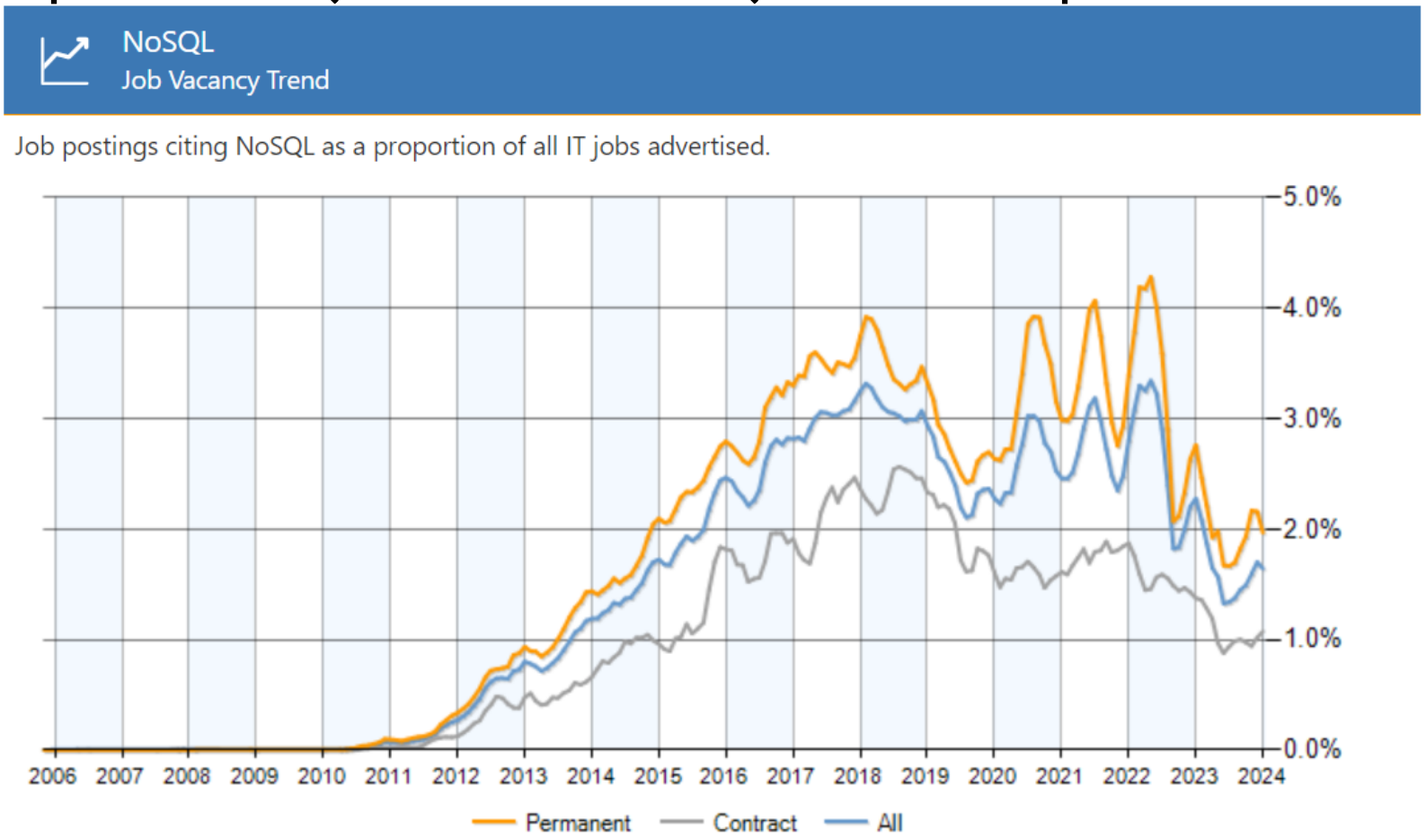
Modelul non relational



Ranking-ul bazelor de date, dupa https://db-engines.com/en/ranking

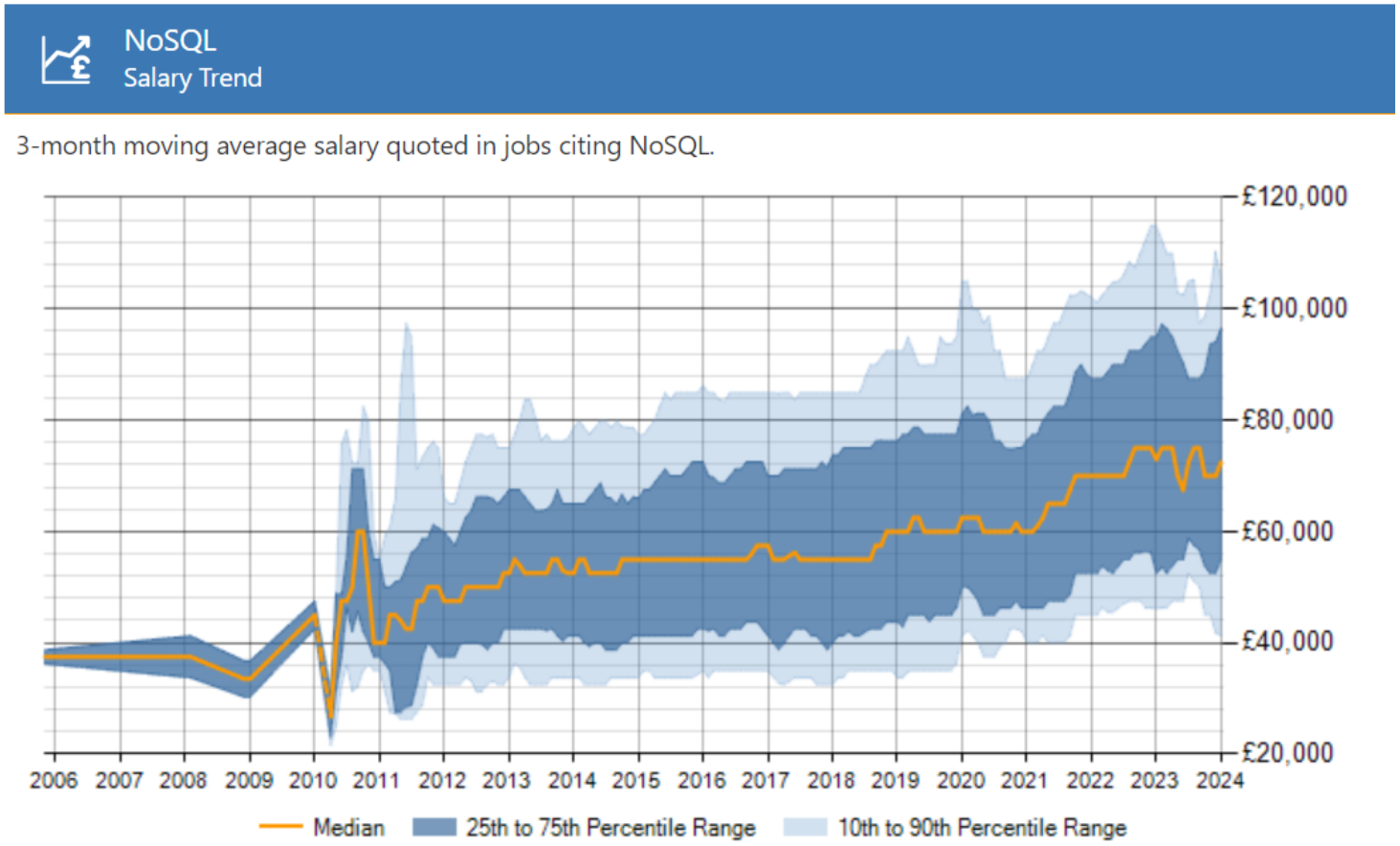
Modelul non relational

- NoSQL job trends, dupa https://www.itjobswatch.co.uk/jobs/uk/nosql.do



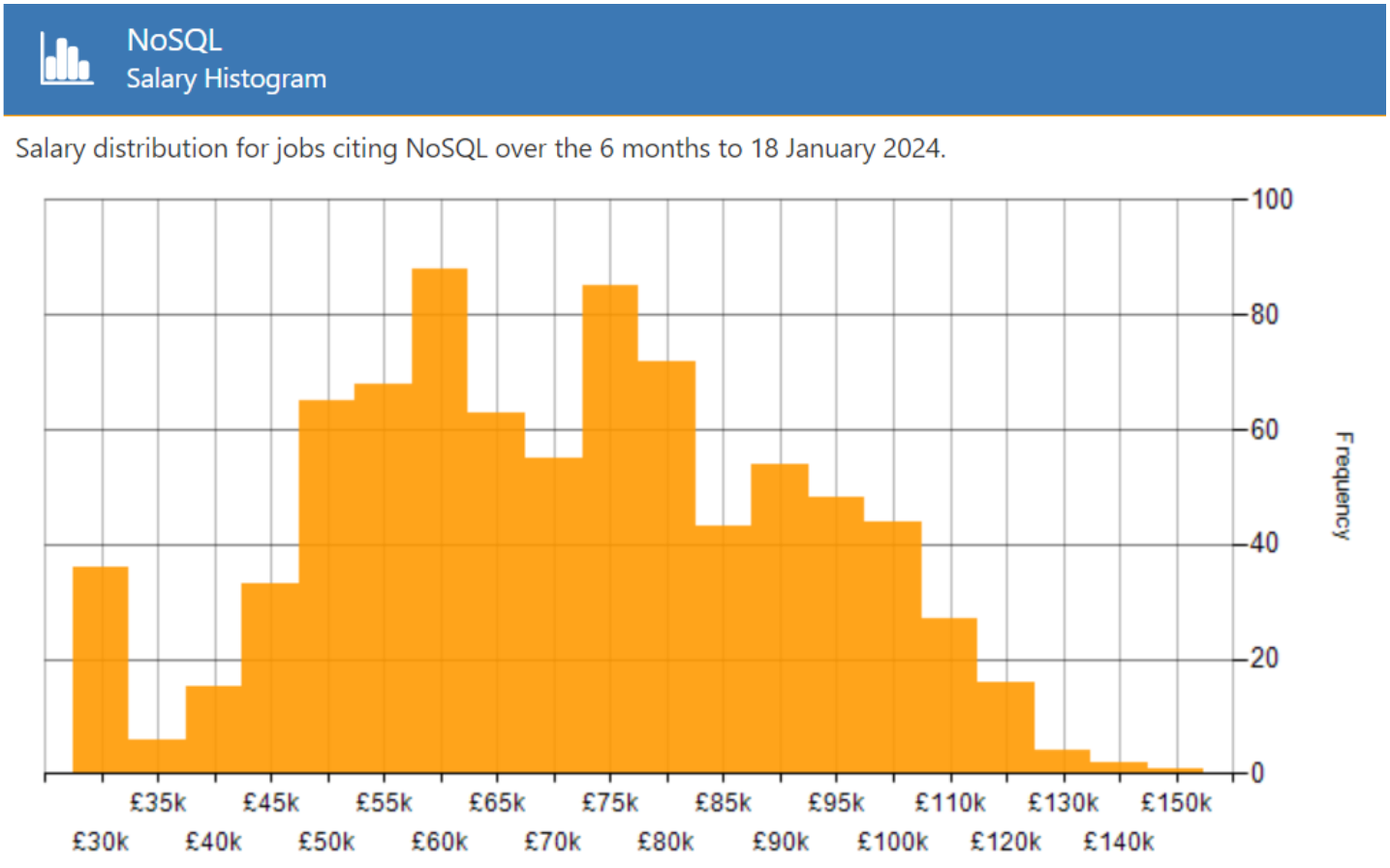
Modelul non relational

- NoSQL salary trends, dupa https://www.itjobswatch.co.uk/jobs/uk/nosql.do



Modelul non relational

- NoSQL salary histogram, dupa https://www.itjobswatch.co.uk/jobs/uk/nosql.do



Modelul non relational

- O clasificare a modelelor NoSQL:

Modelul	Exemple de sisteme
Key-value stores (colecții de perechi cheie-valoare)	Amazon Dynamo, Redis, Membase, MemcacheDB, Scalaris, Tokyo Cabinet, Voldemort, Riak
Column oriented (familii de coloane)	Google Bigtable, Cassandra (Facebook), Hadoop/HBase, HyperTable, Amazon SimpleDB
Document oriented (colectii de documente)	MongoDB, Couchbase, CouchDB, Terrastore, RavenDB
Graph oriented (graf)	Neo4j, InfiniteGraph, InfoGrid, GraghBase, HyperGraphDB

- Lista bazelor de date NoSQL (in present >225) poate fi consultata aici: http://nosql-database.org/

Modelul non relational

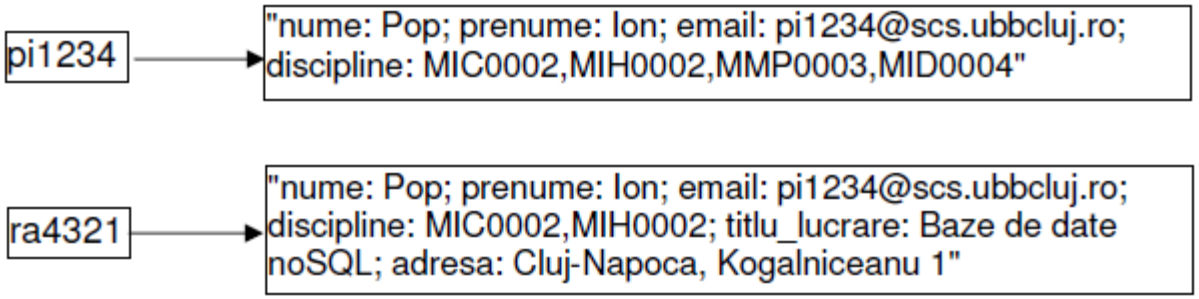
1. **Modelul cheie-valoare** (Key-value stores)

- Baza de date este formata dintr-o **colectie de perechi (cheie, valoare)**
- Cheia si valoarea sunt siruri de caractere, iar cheile sunt distincte (se folosesc pentru identificare)
- Operatiile permise in aceste baze de date sunt:
  - inserarea unei valori pentru o cheie
  - returnarea valorii corespunzătoare unei chei
  - stergerea unei chei impreuna cu valoarea sa

Modelul non relational

1. **Modelul cheie-valoare** (Key-value stores)

- Exemple:



- Memorarea bazei de date se poate face (pentru un acces rapid):
  - prin utilizarea unui "hash table"
  - sub forma unei variante de B-arbore

Modelul non relational

2. Modelul column store (familii de coloane)

- In acest model, o coloană este o pereche cheie-valoare in care cheia este numele coloanei și valoarea este valoarea corespunzătoare coloanei
- Fiecare inregistrate se identifica prin valorile unei chei
- Se recomanda pentru tabele de dimensiune mare, cu multe elemente nedefinite (celule in tabel cu valori null), eventual cu **valori repetitive** (tabele non 1NF)
- In bazele de date relationale valorile coloanelor se memoreaza consecutiv pentru o inregistrare, deci se are acces rapid la toate valorile din inregistrare => **mod de memorare orientat linie**
- In modelul column store se memoreaza consecutiv valorile nenule din fiecare coloana (in cele mai multe cazuri la interogari nu se doresc valorile din toate coloanele) => **mod de memorare orientat coloana**

Modelul non relational

2. Modelul column store (familii de coloane)

- Exemplu:

Presupunem ca avem de memorat urmatorul tabel non1NF (precizat in modelul orientat linie):

ID	nume	prenume	email	contract_studiu
1	Pop	Ion	pi1234@scs.ubbcluj.ro	"MIC0002","MIH0002","MMP0003","MID0004"
2	Popa	Radu		
3	Alb	Ana	aa4321@scs.ubbcluj.ro	"MLR0020", "MLR0002", "MLR5004"

- Memorarea **orientata coloana** este urmatoarea (s-a folosit coloana ID, care este cheia tabelului, pentru fiecare coloana):

ID	nume
1	Pop
2	Popa
3	Alb

ID	prenume
1	Ion
2	Radu
3	Ana

ID	email
1	pi1234@scs.ubbcluj.ro
3	aa4321@scs.ubbcluj.ro

ID	contract_studiu
1	MIC0002
1	MIH0002
1	MMP0003
1	MID0004
3	MLR0020
3	MLR0002
3	MLR5004

Modelul non relational

2. Modelul column store (familii de coloane)

- Exemplu:

- Intr-o celula din tabel se pot pastra mai multe versiuni ale valorii. De exemplu tabelele cu coloana nume si email ar putea sa fie cu urmatorul continut dupa o anumit perioada (s-a mai adaugat o coloana **ts - timestamp**, cu semnificatia: timpul modificarii):

ID	ts	nume
1	t1	Pop
2	t1	Popa
3	t1	Alb
3	t5	Rus

ID	ts	email
1	t1	pi1234@scs.ubbcluj.ro
1	t2	piir1234@scs.ubbcluj.ro
1	t3	pop_ion@yahoo.com
3	t1	aa4321@scs.ubbcluj.ro
3	t4	pop_ana@gmail.com

- Modelul column store poate imparti multimea de coloane intr-un anumit numar de "familii de coloane", intr-o familie fiind incluse coloane aproximativ asemanatoare (folosite impreuna in interogari)
- Exemplu de familii de coloane posibile: {nume, prenume}, {email}, {contract_studiu}

Modelul non relational

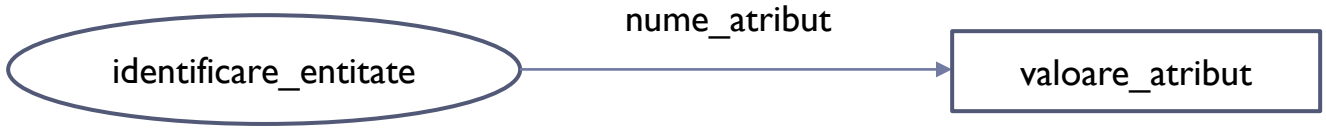
3. Modelul graf

- Presupunem ca avem de memorat o multime de date semistructurate, sau un tabel dintr-o baza de date relationala
- Fiecare entitate (inregistrare) se identifica prin valoarea unei chei

- O astfel de entitate are valori pentru anumite atribute (proprieti, predicate, coloane). Vom construi o multime de **triplete** cu urmatorul continut:

(identificare_entitate, nume_atribut, valoare_atribut)

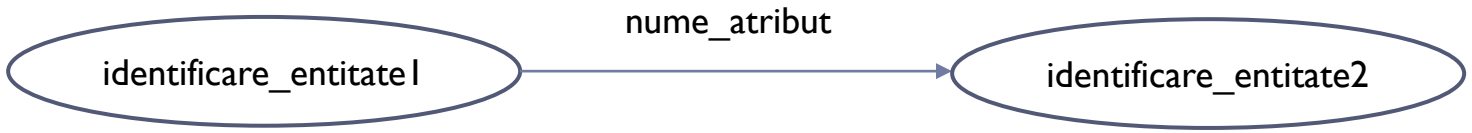
- Cu aceste triplete se poate construi un graf orientat, pentru un triplet apare un arc de forma urmatoare:



Modelul non relational

3. Modelul graf

- Varfurile grafului pot sa fie de doua tipuri:
  - **identificari de entitati** (desenate ca o elipsa), sau
  - **constante** (desenate ca un dreptunghi)
- Valoarea unui atribut poate sa fie o identificare de entitate, si atunci este de primul tip



Modelul non relational

3. Modelul graf

- Exemplu:
- Presupunem ca avem o baza de date relationala cu urmatoarele tabele:

studenti

IDstud	nume	prenume
1	Pop	Ion
2	Popa	Radu
3	Alb	Ana

discipline

cod	denumire
MIC0002	Sisteme de operare distribuite
MIH0002	Baze de date
MMP0003	Probabilităţi şi statistică
MLR0002	Analiză matematică
MLR5004	Arhitectura sistemelor de calcul

contracte

IDstud	coddisc
1	MIC0002
1	MIH0002
1	MMP0003
2	MIH0002
2	MLR5004
3	MLR0002
3	MLR5004

- Tripletele amintite mai sus pentru aceasta baza de date sunt:

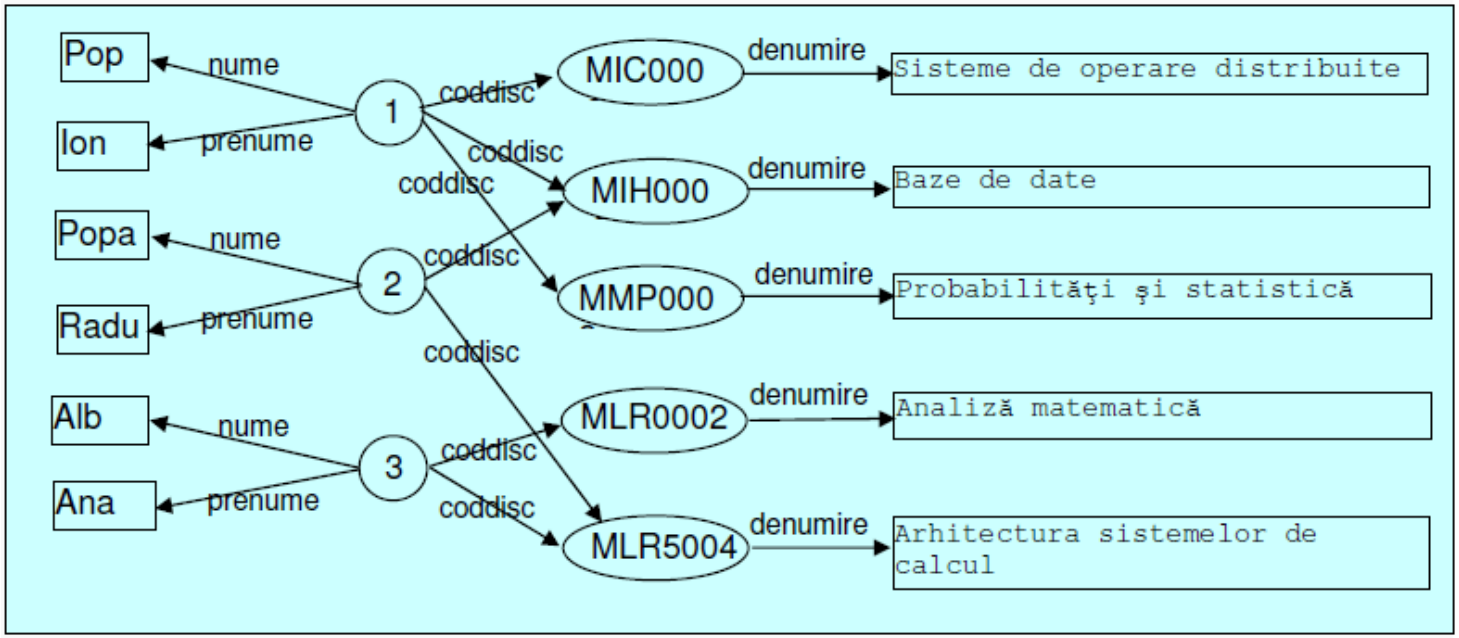
identificare_entitate	nume_atribut	valoare_atribut
1	nume	Pop
1	prenume	Ion
2	nume	Popa
2	prenume	Radu
3	nume	Alb
3	prenume	Ana
MIC0002	denumire	Sisteme de operare distribuite
MIH0002	denumire	Baze de date
MMP0003	denumire	Probabilităţi şi statistică
MLR0002	denumire	Analiză matematică
MLR5004	denumire	Arhitectura sistemelor de calcul
1	coddisc	MIC0002
1	coddisc	MIH0002
1	coddisc	MMP0003
3	coddisc	MLR0002
3	coddisc	MLR5004

Pe coloana "valoare_atribut" s-au marcat cu albastru valorile ce precizeaza identificari de entitate

Modelul non relational

3. Modelul graf

- Exemplu:
- Graful care se poate construi pentru aceste multimi de triplete este urmatorul

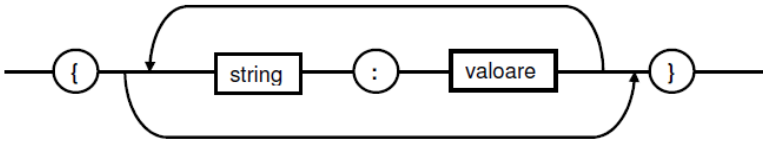


Exemplu de utilizare a acestui mod de memorare la cautarea pe google:
Introducing the Knowledge Graph, http://www.youtube.com/watch?v=mmQl6VGvX-c
The Knowledge Graph, https://www.google.com/intl/es419/insidesearch/features/search/knowledge.html

Modelul non relational

4. **Modelul document** (colectii de documente)

- O baza de date document contine diverse **colectii de documente** (obiecte), **analog tabelelor** dintr-o baza de date relationala
- Intr-o **colectie** se grupeaza **documentele** utile intr-o interogare
- Intre colectii diferite **nu se pot efectua operatii de join**
- **Baza de date** si **colectiile din baza de date** se identifica printr-un **nume** (pentru numele bazei de date nu se poate folosi: admin, local, config)
- Un **document** (obiect) **nu are o structura stabilita**
- Un document are o **identificare** unica in colectia de date, prin campul cu denumirea "**_id**"
- Caracterele (din denumiri, din valori) sunt **case-sensitive**
- Pentru gestiunea unei baze de date si a unei colectii exista **mai multe metode**
- Un **document** este format dintr-o multime de perechi (campuri) **nume/valoare**, precizat sub forma urmatoare:



Modelul non relational

4. **Modelul document** (colectii de documente)

- Structura de document se foloseste pentru:
  - documentele memorate in baza de date
  - precizarea conditiilor pe care le indeplinesc documentele utile intr-o comanda (citire, actualizare, stergere)
  - precizarea tipurilor de modificari pentru documente
  - specificarea modului de construire a indexurilor
  - diverse optiuni in comenzi

Modelul non relational

4. Modelul document (colectii de documente)

- **numele** din pereche (camp) este un sir de caractere unic in multimea de perechi de la un document
  - Acest sir se poate delimita cu apostroafe sau ghilimele (depinde si de limbajul folosit pentru gestiune)
  - Formatul acesta este **JSON** (JavaScript Object Notation) si constituie o alternativa (cu mai putine facilitati) pentru **XML**
  - Exista multe clase (biblioteci) sau functii pentru gestiunea acestor documente in diverse limbaje de programare

Modelul non relational

4. **Modelul document** (colectii de documente)

- **valoarea** din pereche poate fi de unul din urmtoarele tipuri:
  - un s**ir de caractere** delimitat cu apostroafe sau ghilimele
  - un **num**ar
    - Valoarea implicit va fi flotant (reprezentat n virgul flotant). Pentru numere intregi se pot folosi reprezentrile pe 4 sau 8 octeti (cu constructori pentru aceste tipuri)
  - o **colectie** (array) de valori de acelasi tip, colectia e delimitata de "[" si "]" si valorile sunt separate cu **","**
  - tipuri corespunzatoare diverselor implementari (booleene, data calendaristica, date binare, etc.)
  - un **document** de acest tip (deci se poate face o imbricare a valorilor)

Modelul non relational

4. Modelul document (colectii de documente)

- **Fiecare document** are o **identificare**, care este valoarea pentru campul cu numele **"_id"**(deci un camp cu acest denumire nu poate fi folosit pentru alte scopuri)
- Valoarea pentru acest camp este precizata (explicit sau implicit) la inserarea documentului
- Valorile acestui camp trebuie sa fie distincte pentru documentele unei colectii
- Daca valoarea nu este precizata explicit, atunci se genereaza automat

Modelul non relational

4. Modelul document (colectii de documente)

Exemplu de document dintr-o bază de date de acest tip:

```
{
  "nume": "Pop",
  "prenume": "Ion",
  "contract_studiu": ["MIC0002","MIH0002", "MMP0003", "MID0004"],
  "adresa": {
    "localitatea": "Cluj-Napoca",
    "strada": "Kogalniceanu",
    "numarul": 1
  }
  "email": "pi1234@scs.ubbcluj.ro"
}
```

In continuare se vor da câteva exemple de memorare a unor informaţii.

Exemplul 1: Memorarea relaţiilor 1:1

Presupunem că trebuie memorate unele informaţii despre **persoane** şi **adresele** acestora. Informaţiile se pot păstra:

a. în documente diferite (ca într-o bază de date normalizată):

```
{
 _id: "pibd1234",
 "nume": "Pop Ion",
 "grupa": "243",
 "email": ["pibd1234@scs.ubbcluj.ro", "pop_ion@yahoo.com"]
}
{
```

Modelul non relational

4. Modelul document (colectii de documente)

```
"titular": "pibd1234",
 "localitatea": "Cluj-Napoca",
 "strada": "Kogalniceanu",
 "nr": 1
}
```

b. în acelaşi document (variantă utilă dacă adresa se foloseşte des împreună cu alte date despre persoane):

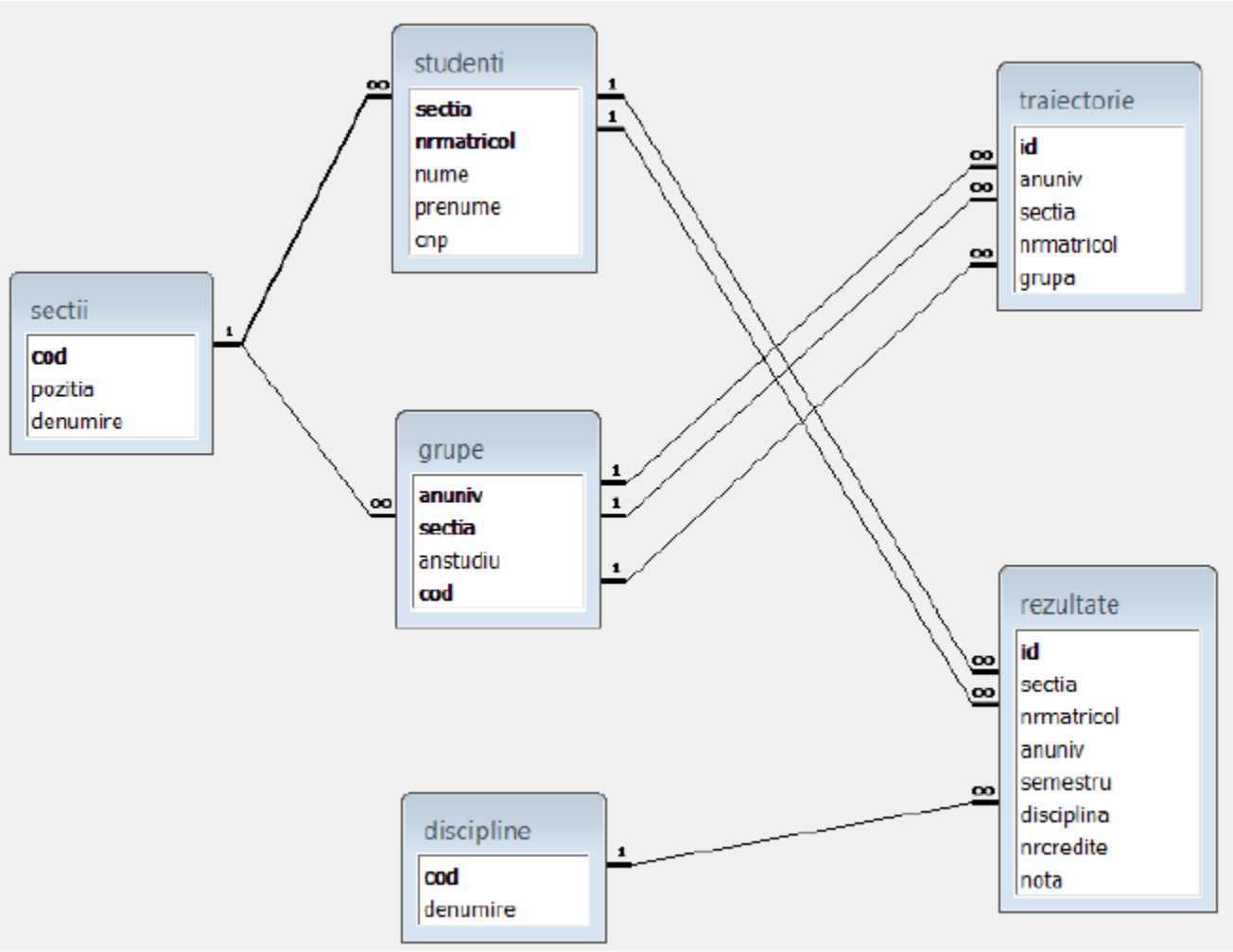
```
{
 "nume": "Pop Ion",
 "grupa": "243",
 "email": ["pibd1234@scs.ubbcluj.ro", "pop_ion@yahoo.com"],
 "adresa":
    {"titular": "pibd1234",
     "localitatea": "Cluj-Napoca",
     "strada": "Kogalniceanu",
     "nr": 1
    }
}
```

Exemplul 2: Memorarea relaţiilor 1:n şi m:n

Presupunem că trebuie memorate unele informaţii despre studenţi dintr-o facultate (informaţiile pot să fie deja memorate într-o bază de date normalizată, cu schema dată în figura următoare).

Modelul non relational

4. Modelul document (colectii de documente)



Modelul non relational

4. Modelul document (colectii de documente)

Aceste informaţii se pot păstra:

a. fiecare înregistrare din aceste tabele are un document în acest model. Exemplu pentru unele câmpuri din câteva documente

```
{"_id": "bd",
 "denumire": "Baze de date",
}
{_id: "prbd1235",
 "nume": "Pop",
 "prenume": "Radu",
 "sectia": "bd",
}
{_id: "pabd1233",
 "nume": "Pop",
 "prenume": "Ana",
 "sectia": "bd",
}
```

b. în documente diferite cu memorarea identificărilor studenţilor de la o specializare:

```
{"_id": "bd",
 "denumire": "Baze de date",
 "studenti": ["prbd1235","pabd1233"]
}
{_id: "prbd1235",
 . . .
```

Modelul non relational

4. Modelul document (colectii de documente)

```
}
{_id: "pabd1233",
. . .
}
```

c. în acelaşi document

```
{ "_id": "mrbd0420",
  "cnp": "1791216042000",
  "nrmatricol": "420",
  "nume": "Moldovan",
  "prenume": "Radu-Marius",
  "rezultate": [
    {
      "anuniv": "2011",
      "semestru": NumberInt(1),
      "cod-disciplina": "MI293",
      "denumire-disciplina": "Gestiunea proiectelor",
      "nrcredite": 9,
      "nota": NumberInt(8)
    },
    {
      "anuniv": "2011",
      "semestru": NumberInt(2),
      "cod-disciplina": "MI355",
      "denumire-disciplina": "Securitatea in Internet",
      "nrcredite": 9,
      "nota": NumberInt(8)
```

Modelul non relational

4. Modelul document (colectii de documente)

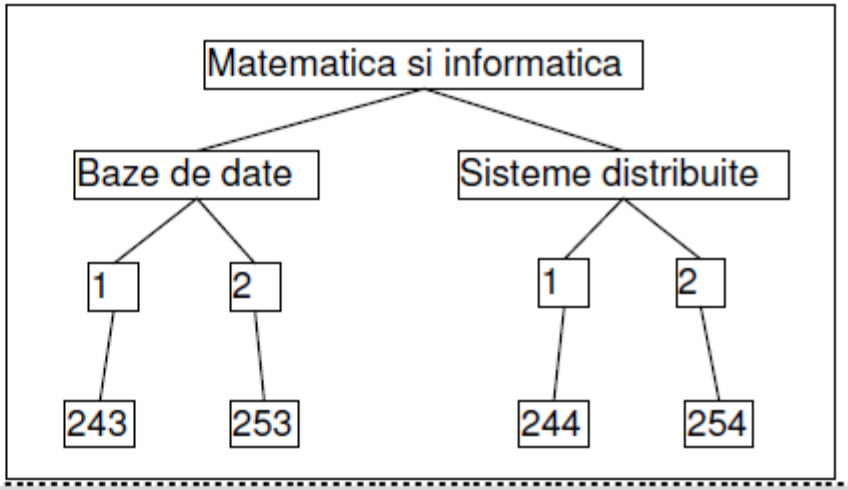
```
    }
  ],
  "sectia": "Baze de date",
  "traiectorie": [
    {
      "anuniv": "2011",
      "grupa": "243",
      "anstudiu": NumberInt(1)
    },
    {
      "anuniv": "2012",
      "grupa": "253",
      "anstudiu": NumberInt(2)
    }
  ]
}
```

Modelul non relational

4. Modelul document (colectii de documente)

Exemplul 3: Memorarea structurilor arborescente

Presupunem că trebuie memorate unele informaţii despre formaţiile de studiu (secţii, ani de studiu, grupe) dintr-o facultate. Câteva dintre aceste informaţii apar în figura următoare.



Aceste informaţii se pot păstra:

a. în documente diferite, conform unei baze de date relaţionale în 3NF.

b. Memorarea structurii arborescente cu **precizarea referinţei la nodul părinte**:

```
{"_id": "fac_mi", "denumire": "Matematica si informatica", "parinte": null}
{"_id": "spec_bd", "denumire": "Baze de date", "parinte": "fac_mi"}
{"_id": "spec_sd", "denumire": "Sisteme distribuite", "parinte": "fac_mi"}
{"_id": "an_bd_1", "denumire": "1", "parinte": "spec_bd"}
{"_id": "an_bd_2", "denumire": "2", "parinte": "spec_bd"}
{"_id": "an_sd_1", "denumire": "1", "parinte": "spec_sd"}
{"_id": "an_sd_2", "denumire": "2", "parinte": "spec_sd"}
{"_id": "gr_bd_243", "denumire": "243", "parinte": "an_bd_1"}
{"_id": "gr_bd_253", "denumire": "253", "parinte": "an_bd_2"}
{"_id": "gr_sd_244", "denumire": "244", "parinte": "an_sd_1"}
{"_id": "gr_sd_254", "denumire": "254", "parinte": "an_sd_2"}
```

Modelul non relational

4. Modelul document (colectii de documente)

Pentru a determina documentul părinte se foloseşte câmpul "parinte", iar pentru a determina nodurile fiu ale unui document se caută documentele pentru care câmpul "parinte" are o valoare dată.

c. Memorarea structurii arborescente cu precizarea referinţelor la nodurile "copil":

```
{"_id": "fac_mi", "denumire": "Matematica si informatica",
    "copii": ["spec_bd", "spec_sd"]}
{"_id": "spec_bd", "denumire": "Baze de date",
    "copii": ["an_bd_1", "an_bd_2"]}
{"_id": "spec_sd", "denumire": "Sisteme distribuite",
    "copii": ["an_sd_1", "an_sd_2"]}
{"_id": "an_bd_1", "denumire": "1", "copii": ["gr_bd_243"]}
{"_id": "an_bd_2", "denumire": "2", "copii": ["gr_bd_253"]}
{"_id": "an_sd_1", "denumire": "1", "copii": ["gr_sd_244"]}
{"_id": "an_sd_2", "denumire": "2", "copii": ["gr_sd_254"]}
{"_id": "gr_bd_243", "denumire": "243", "copii": []}
{"_id": "gr_bd_253", "denumire": "253", "copii": []}
{"_id": "gr_sd_244", "denumire": "244", "copii": []}
{"_id": "gr_sd_254", "denumire": "254", "copii": []}
```

Referinţele la nodurile fiu se determină repede cu valoarea câmpului "copii", iar nodul părinte se determină prin căutarea documentului care conţine o valoare dată în câmpul "copii".

Modelul Relational – Modelul non Relational. Analiza comparativa

Caracteristici	NoSql	SQL
Stocarea datelor	- Documente (JSON) - Flexibila - Nu e nevoie ca fiecare inregistrare sa stocheze aceleasi proprietati/atribute - Proprietati/atribute noi se pot adauga dinamic - Adecvat pentru date semi-structurate si nestructurate	- Tabele ale BD - Mai putin flexibila - Impune ca toate inregistrarile sa aibe aceleasi attribute - Adaugarea unui nou atribut va atrage dupa sine modificarea schemei - Adecvat pentru date structurate
Schema	- Permite scheme flexibile sau dinamice - Aplicatia e cea care dicteaza schema	- Permite doar scheme stricte - Schema trebuie sa fie sincronizata intre aplicatie si baza de date
Tranzactii	- Suportul tranzactiilor ACID depinde de la solutie la solutie	- Tranzactii ACID
Consistenta si Disponibilitate	- Depinde de solutie - Consistenta, disponibilitatea si performanta pot fi decise in functie de nevoile aplicatiei	- Consistenta puternica - Consistenta primeaza in fata disponibilitatii datelor si a performantei

Examen scris

- I. Subiect teoretic – 4p
  - Algoritm/metoda, eventual aplicat/a
  - O definitie
  - Explicarea unui/unor concept/e si exemple
- II. Problema – 5p
  - Crearea une baze de date in 3NF
  - 2 interogari
    - Una folosind algebra relationala
    - Una folosind limbajul SQL
- Oficiu – 1p

Bibliografie

- Leon Tambulea, Curs de Baze de Date, UBB Cluj
- G. Cosofret, I. Ciuciu, Superstore Sales Reporting: A Comparative Analysis of Relational and Non-relational Databases, OTM 2017 Workshops, Springer LNCS, vol. 10697, chapter 10
- [Codd70], E., A Relational Model of Data for Large Shared Data Banks, IBM Research Laboratory, Communications of the ACM, vol. 13 (6) (1970)
- [GiLy02] Seth Gilbert, Nancy Lynch, Brewer's conjecture and the feasibility of consistent, available, partition-tolerant web services, ACM SIGACT News, Volume 33 Issue 2, June 2002, Pages 51-59, sau: http://lpd.epfl.ch/sgilbert/pubs/BrewersConjecture-SigAct.pdf
- [Hurst] Nathan Hurst, Visual Guide to NoSQL Systems, http://blog.nahurst.com/visual-guide-to-nosql-systems
- [JSON] Introducing JSON, http://www.json.org/
- [NoSQL] Guide to theNon - Relational Universe, http://nosql-database.org/
- [PeTi] Perdue, Tim, NoSQL: An Overview of NoSQL Databases, http://newtech.about.com/od/databasemanagement/a/Nosql.htm