# Gemixque

## Sistem de recomandări de jocuri video

Radu Damian

Dr. Cristian Frăsinaru

Facultatea de Informatică

2022

# Cuprins

# Privirea în ansamblu a problemei

Trei elemente principale:

- Utilizatorul
- Jocul video
- Recenzia

# Specificații

Fie $U$ mulțimea de utilizatori și $G$ mulțimea de jocuri.

$$\forall u \in U,\ \exists\ G_u \subseteq G$$

$$\forall g \in G_u,\ s(u, g) = r_{ug}$$

$$1 \leq r_{ug} \leq 10$$

# Obiectiv

- Fie $u \in U$, $g \notin G_u$
- $\hat{s}(u, g) = ?$

# Introducere în Neo4j

- Bază de date NoSQL de tip graf
- Datele sunt reținute prin intermediul nodurilor și muchiilor
- Utilizează limbajul de interogare Cypher

# Caracteristici

Un nod are următoarele trăsături:

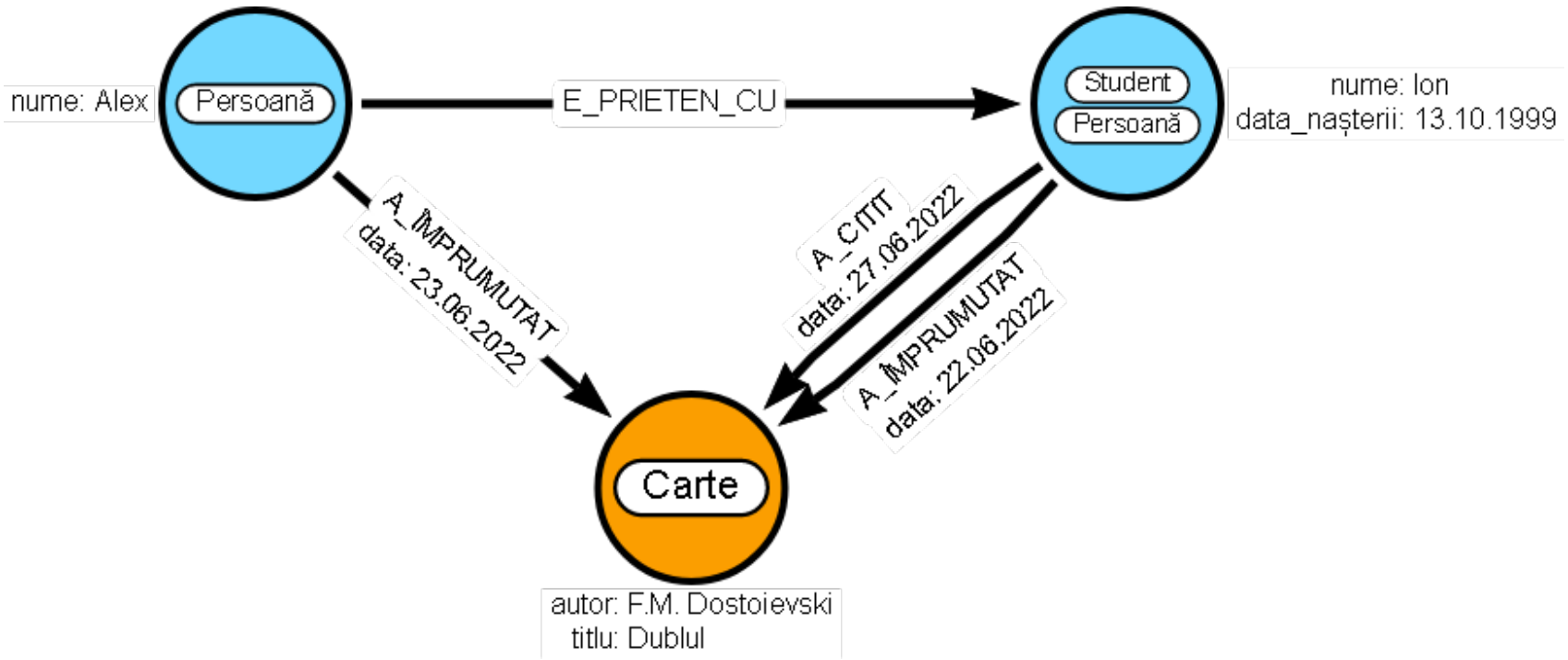
- Reprezintă obiecte/entități
- Poate fi etichetat
- Poate avea proprietăți

# Caracteristici

O muchie(relație) este caracterizată de:

- Tipul acesteia
- Direcția
- Poate avea proprietăți

# Exemplu


nume: Alex
Persoană
E_PRIETEN_CU
Student
Persoană
nume: Ion
data_nașterii: 13.10.1999
A_ÎMPRUMUTAT
data: 23.06.2022
A_CITIT
data: 27.06.2022
A_ÎMPRUMUTAT
data: 22.06.2022
Carte
autor: F.M. Dostoievski
titlu: Dublul

# Limbajul de interogare Cypher

- inspirat din SQL
- prezintă un mod intuitiv de reprezenta noduri și relații prin intermediul unei sintaxe tip ASCII-art

# Cypher vs. SQL

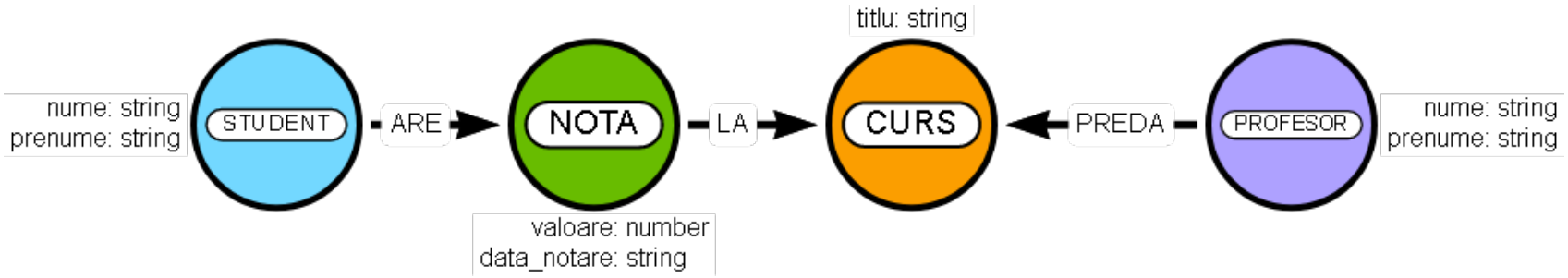
Studiu de caz:

- modelarea situației din cadrul unei facultăți(studenți, cursuri, profesori)
- exemplu interogare în SQL
- exemplu interogare în Cypher
- conceptul de *pattern-matching*

# Cypher vs. SQL

```
SELECT p.nume, p.prenume FROM NOTE n
JOIN CURSURI c ON n.id_curs = c.id
JOIN DIDACTIC d ON d.id_curs = c.id
JOIN PROFESORI p ON p.id = d.id_profesor
WHERE VALOARE = 10 AND ID_STUDENT = 36;
```

# Cypher vs. SQL



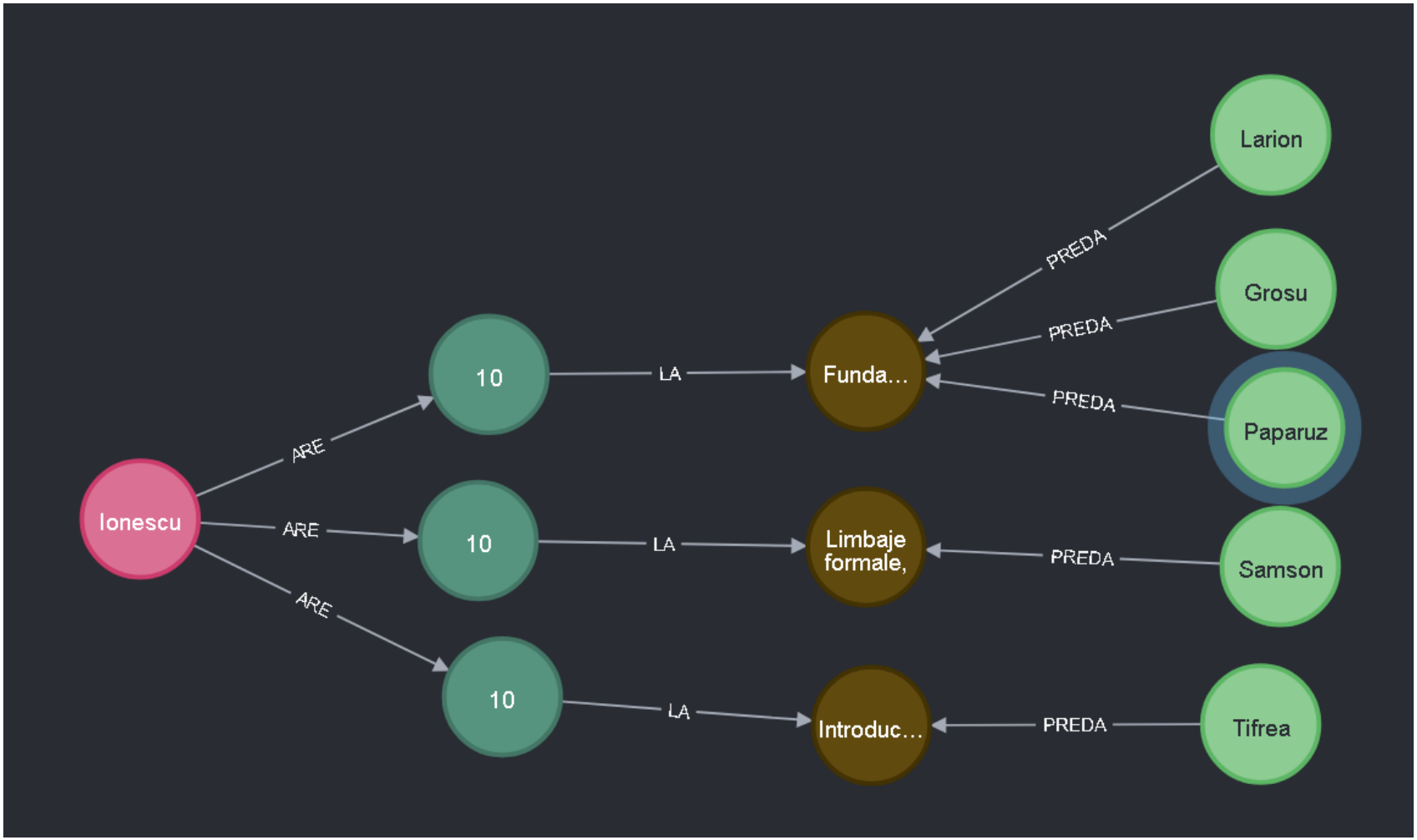
# Cypher vs. SQL

```
MATCH (s:STUDENT)-[:ARE]->(n:NOTA {valoare: 10}),
(n)-[:LA]->(:CURS)<-[:PREDA]-(p:PROFESOR)
WHERE id(s) = 0
RETURN p.nume, p.prenume
```

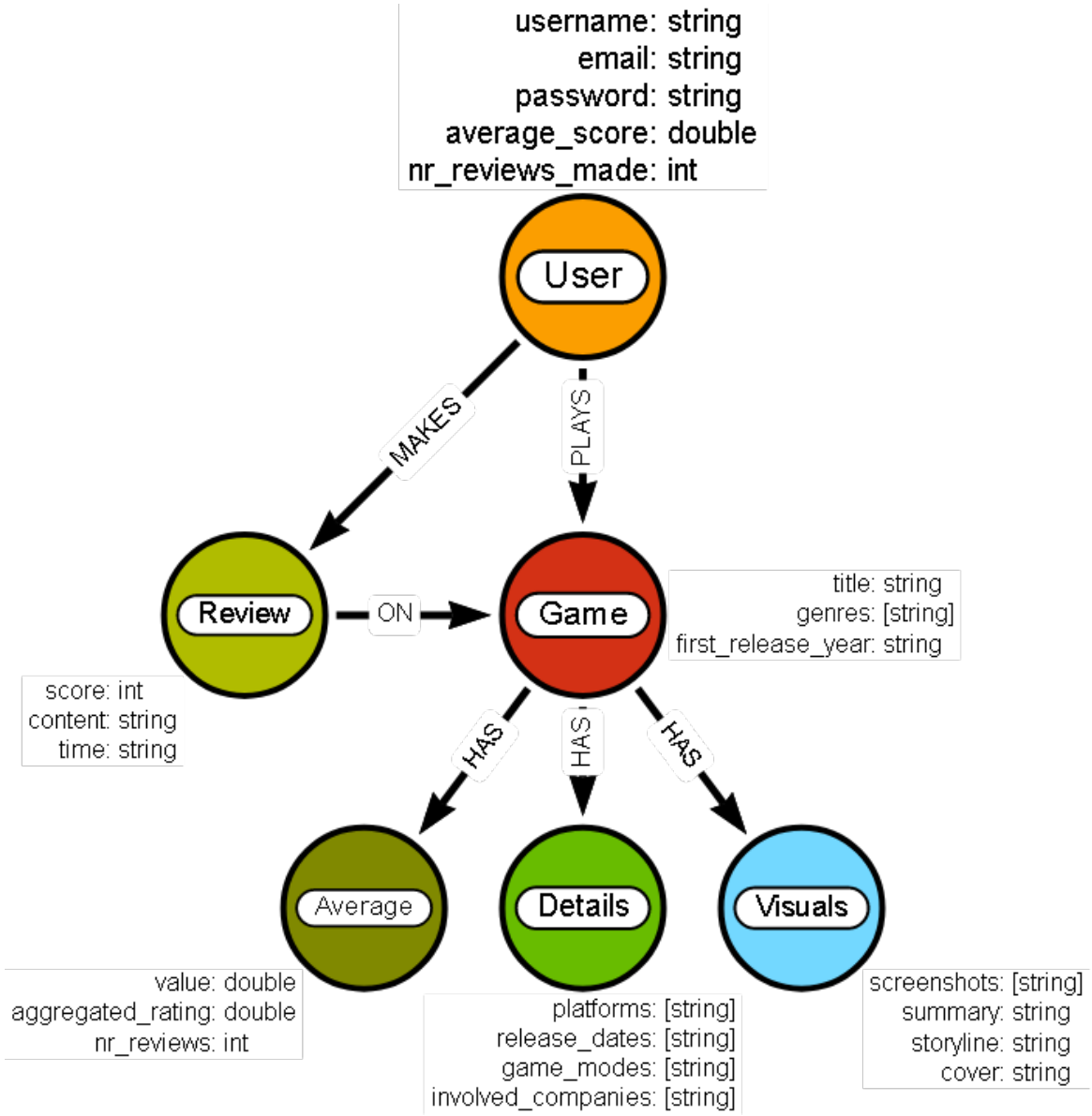
# Pattern-matching

- Conceptul de *pattern-matching* este introdus prin intermediul clauzei `MATCH`
- În funcție de cum este definit *pattern*-ul în interogare, graful va fi parcurs într-un anumit mod

# Exemplu


Ionescu
ARE
ARE
ARE
10
10
10
LA
LA
LA
Funda...
Limbaje formale,
Introduc...
PREDA
PREDA
PREDA
PREDA
PREDA
Larion
Grosu
Paparuz
Samson
Tifrea

# Schema bazei de date



# Introducere

- Ipoteză: cantitate masivă de informații
- Problemă: filtrarea acestor informații
- Soluțe: sistem de recomandare ce oferă conținut personalizat utilizatorilor

# Sistem de recomandări bazat pe filtrare colaborativă

- resurse ce nu pot fi descrise prin metadate cu ușurință
- matrice de scoruri utilizator-resursă
- calcularea unei predicții pentru elementele lipsă din matrice

# Exemplu

| | DOOM Eternal | Battlefield | Call of Duty | The Witcher 3: Wild Hunt | Dark Souls III |
|---|---|---|---|---|---|
| Ion | 9 | 8 | 10 | 3 | 6 |
| Gigel | 8 | 9 | ? | 4 | 5 |
| Alex | 5 | 7 | 4 | 10 | 9 |

# Formule

$$s(u,g) = \frac{\sum_{u' \epsilon \Omega_g} w_{uu'} \cdot r_{u'g}}{\sum_{u' \epsilon \Omega_g} w_{uu'}}$$

$\Omega_g$ - mulțimea utilizatorilor care au atribuit un scor jocului $g$
$r_{ug}$ - nota oferită de utilizatorul $u$ jocului g
$w_{uu'}$ - gradul desimilaritate între doi utilizatori

# Formule

$$dev(u, g) = r_{ug} - \overline{r_u}$$

$$dev(\hat{u}, g) = \frac{1}{|\Omega_g|} \cdot \sum_{u' \epsilon \Omega_g} dev(u', g)$$

$\overline{r_u}$ - media scorurilor utilizatorului $u$

# Formule

$$\hat{w\_dev}(u,g) = \frac{\sum_{u' \epsilon \Omega_g} w_{uu'} \cdot dev(u',g)}{\sum_{u' \epsilon \Omega_g} |w_{uu'}|}$$

$$\hat{s}(u,g) = \overline{r_u} + \hat{w\_dev}(u,g) \quad (1)$$

# Coeficientul lui Pearson

$$w_{uu'} = \frac{\sum_{g \epsilon \Psi_{uu'}} dev(u,g) \cdot dev(u',g)}{\sqrt{\sum_{g \epsilon \Psi_{uu'}} dev(u,g)^2} \cdot \sqrt{\sum_{g \epsilon \Psi_{uu'}} dev(u',g)^2}}$$

$\Psi_{uu'}$ - mulțimea jocurilor în comun recenzate de utilizatorii $u$ și $u'$
$\Psi_u$ - mulțimea jocurilor recenzate de utilizatorul $u$
$\Psi_{uu'} = \Psi_u \cap \Psi_{u'}$

# Algoritm - pseudocod

$u \in U, k \in \mathbb{N}$

getRecommendations$(u, k)$

$S \leftarrow \varnothing$

*submulțime de k utilizatori similari cu u ordonați descrescător după pondere:*

$|U|^k = \{X \mid X \subseteq U \wedge |X| = k \wedge \text{argmax}_X \sum_{x \in X} w_{ux}\}$

**for** $u' \in |U|^k$ **do**

*jocuri recenzate de u' care nu sunt în comun cu jocurile recenzate de u:*

$G' = \Psi_{u'} \setminus \Psi_{uu'}$

**for** $g' \in G'$ **do**

$S = S \cup \{g'\}$

actualizare scor $g'$ utilizând formula (1)

**end for**

**end for**

# Concluzii

- cu ajutorul Neo4j se pot rezolva probleme în care relațiile dintre entități au o importanță semnificativă
- algoritmul de recomandare prezentat reprezintă un punct de pornire în dezvoltarea în amănunt a unui sistem de recomandare