Grafowe bazy danych
Baza gry karcianej

Krystian Gronkowski, indeks: 281184

Maj 2024

Hasło do połączenia się z bazą danych online:

```
# Wait 60 seconds before connecting using these details, or login to https://console.neo4j.io
      to validate the Aura Instance is available
NEO4J_URI=neo4j+s://fea71034.databases.neo4j.io
NEO4J_USERNAME=neo4j
NEO4J_PASSWORD=TrHt25o99lYPPNztD0xyAceH_7BnzGBOY99Ytb_BFkw
AURA_INSTANCEID=fea71034
AURA_INSTANCENAME=Instance01
```

Importowanie bazy

types csv:

```
LOAD CSV WITH HEADERS FROM 'file:///types.csv' AS row MERGE (type :Type{Id:
    toInteger(row.Id)}) SET type.Type = row.Type
```

cards csv:

```
LOAD CSV WITH HEADERS FROM 'file:///cards.csv' AS row
MERGE (card:Card {Id: toInteger(row.Id)})
SET card.Name = row.Name,
    card.Effect = row.Effect,
    card.LVL = toInteger(row.LVL),
    card.ATK = toInteger(row.ATK),
    card.DEF = toInteger(row.DEF),
    card.Price = toFloat(row.Price)
WITH card, row
MATCH (type:Type {Id: toInteger(row.Card_Type)})
MERGE (card)-[:TYPE]->(type);
```

decks csv:

```
LOAD CSV WITH HEADERS FROM 'file:///decks.csv' AS row MERGE (deck :Deck{Id:
    toInteger(row.Id)}) SET deck.Name = row.Name, deck.Winrate = toInteger(row.
    winrate), deck.TournamentWins = toInteger(row.tournament_wins)
```

card to deck csv:

```
LOAD CSV WITH HEADERS FROM 'file:///card_to_deck.csv' AS row
MATCH (card :Card{Id: toInteger(toInteger(row.card_id))})
WITH card, row
MATCH (deck :Deck{Id:toInteger(toInteger(row.deck_id))})
MERGE (card)-[:IN_DECK]->(deck)
```

user csv:

```
LOAD CSV WITH HEADERS FROM 'file:///users.csv' AS row MERGE (user :User{Id:
    toInteger(row.Id)}) SET user.Name = row.Name, user.ShowCollection = toBoolean(
    toInteger(row.show_collection))
```

user friends csv:

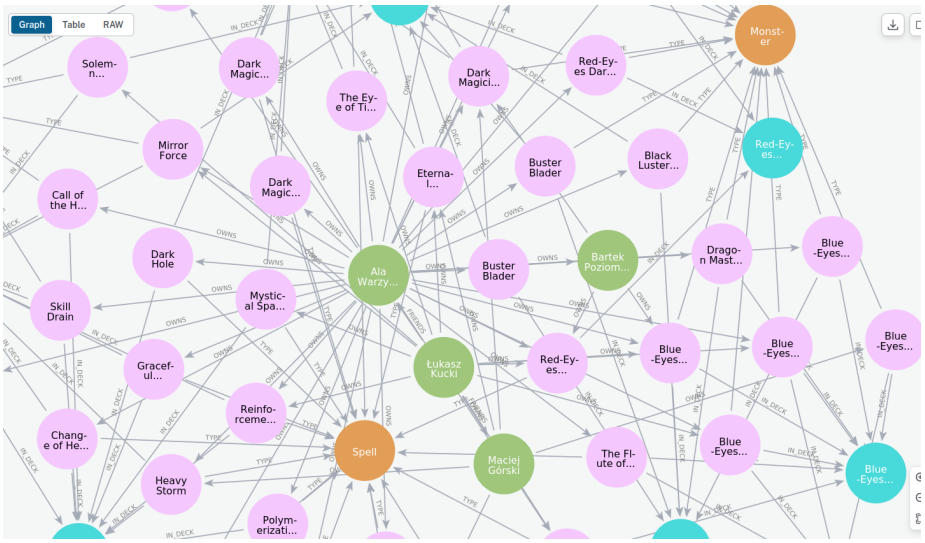
```
LOAD CSV WITH HEADERS FROM 'file:///user_friends.csv' AS row
MATCH (user :User{Id:toInteger(toInteger(row.user_id))})
WITH user, row
MATCH (friend :User{Id:toInteger(toInteger(row.friend_id))})
MERGE (user)−[:FRIENDS]−(friend)
```

cards to user csv:

```
LOAD CSV WITH HEADERS FROM 'file:///card_to_user.csv' AS row
MATCH (user :User{Id:toInteger(row.user_id)})
WITH user, row
MATCH (card :Card{Id:toInteger(row.card_id)})
MERGE (user)−[:OWNS]−>(card)
```

Wyświetlanie zimportowanej bazy danych:

```
MATCH(n) OPTIONAL MATCH(n)−[r]−() RETURN n,r
```



1. Zarządzanie własną kolekcją:

Wyświetl wszystkie karty posiadane przez użytkownika (należy zauważyć, że tylko karty typu "Monster"mają ATK, DEF i LVL. Więc wyświetlamy je tylko jeżeli karta jest potworem, jeżeli jest zaklęciem lub pułapką zamiast tego wyświetlamy napis "Spell"):

```
MATCH (u:User {Name: "Łukasz Kucki"})−[:OWNS]−>(c:Card)
OPTIONAL MATCH (c)−[:TYPE]−>(tCard_Type {Type: 'Monster'})
RETURN
  c.Name AS card_name,
  c.Effect AS effect,
  CASE
    WHEN t IS NOT NULL THEN c.LVL
    ELSE "SPELL"
  END AS level,
  CASE
    WHEN t IS NOT NULL THEN c.ATK
    ELSE "SPELL"
  END AS attack,
  CASE
    WHEN t IS NOT NULL THEN c.DEF
    ELSE "SPELL"
  END AS defense,
  c.Price AS price
```

card_name	effect	level	attack	defense	price
"Blue-Eyes White Dragon"	"This legendary dragon is a powerful engine of destruction. Virtually invincible, very few have faced this awesome creature and lived to tell the tale."	"8"	3000	2500	1.58
"Ultimate Fusion"	"During the Main Phase: Fusion Summon 1 Fusion Monster from your Extra Deck that mentions "Blue-Eyes White Dragon" or "Blue-Eyes Ultimate Dragon" as material	"SPELL"	"SPELL"	"SPELL"	5.32

Wyświetl liczbę posiadanych kart przez użytkownika, oraz ich wartość:

```
MATCH (u:User {Name: "Ala Warzyńska"})−[:OWNS]−>(c:Card)
RETURN u.Name AS user_name, COUNT(c) AS total_cards_owned,SUM(c.Price) AS
      total_value_of_cards_owned
```

Zrób to również dla wszystkich jego znajomych, wstaw "Collection Hidden", jeżeli kolekcja jest ukryta:

```
MATCH (u:User {Name: 'Łukasz Kucki'})−[:OWNS]−>(c:Card)
WITH u, COUNT(c) AS total_cards_owned, SUM(c.Price) AS
      total_value_of_cards_owned
```

```
RETURN u.Name AS user_name, total_cards_owned, total_value_of_cards_owned
ORDER BY total_value_of_cards_owned DESC
LIMIT 1

UNION

MATCH (u:User {Name: 'Łukasz Kucki'})-[:FRIENDS]->(f:User)
OPTIONAL MATCH (f)-[:OWNS]->(fc:Card)
WITH f,
    CASE WHEN f.ShowCollection THEN COUNT(fc) ELSE 'Collection Hidden' END
        AS total_cards_owned,
    CASE WHEN f.ShowCollection THEN SUM(fc.Price) ELSE 'Collection Hidden' END
        AS total_value_of_cards_owned
RETURN f.Name AS user_name, total_cards_owned, total_value_of_cards_owned
ORDER BY total_value_of_cards_owned DESC
```

user_name	total_cards_owned	total_value_of_cards
"Łukasz Kucki"	11	76.48
"Maciej Górski"	5	95.62
"Ala Warzyńska"	"Collection Hidden"	"Collection Hidden"

2. Kompletowanie talii

Wyświetl użytkownikowi nieskompletowane talie, ich statystyki, ile kart mu brakuje do skompletowania talii i ile kosztowałoby jej skompletowanie.

```
MATCH (u:User {Name: 'Bartek Poziomka'})
MATCH (d:Deck)<-[:IN_DECK]-(c:Card)
OPTIONAL MATCH (u)-[o:OWNS]->(c)
WITH u, d, c, COUNT(o) AS owns_card
WHERE owns_card = 0
WITH d, COLLECT(c) AS missing_cards, SUM(c.Price) AS cost_to_complete
RETURN d.Name AS deck_name, d.Winrate AS deck_winrate, d.TournamentWins AS
    tournament_wins, SIZE(missing_cards) AS missing_cards_count, cost_to_complete
ORDER BY cost_to_complete DESC
```

deck_name	deck_winrate	tournament	missing	cost_to_
"Blue-Eyes Starter"	68	3	6	185.290
"Dark Magician Chronicles"	72	7	11	169.77
"Phantom's Madness"	62	1	15	168.150
"Magician's Force"	66	2	10	166.39
"Chaos Masters"	38	0	9	158.94
"Kaiba Duelist Deck"	51	0	10	153.35

3. Polecanie zakupu - Najczęściej używane karty

Wyświetl użytkownikowi wszystkie karty których nie posiada wraz z ich ceną i ilością talii w których występują. Posegreguj malejąco ilością występowania - te w największej liczbie talii są najbardziej warte zakupu.

```
MATCH (u:User {Name: 'Łukasz Kucki'})
MATCH (c:Card)−[:IN_DECK]−>(d:Deck)
WHERE NOT (u)−[:OWNS]−>(c)
WITH c, COUNT(d) AS decks_count
RETURN c.Name AS card_name, decks_count, c.Price AS card_price
ORDER BY decks_count DESC
```

card_name	decks_count	card_price
"Pot of Greed"	9	106.69
"Dragon Master Knight"	4	7.66
"Black Luster Soldier - Envoy of the Beginning"	3	5.21
"Skill Drain"	3	3.0

4. Polecanie zakupu - Uzupełnianie talii

Dla każdej talii, wyświetl użytkownikowi karty z tej talii których nie posiada. Posegreguj wyniki w taki sposób, żeby talie z najmniejszą ilościa brakujących kart wyświetlały swoje rekomendacje na samej górze. Kupując karty polecane w ten sposób, użytkownik stworzyłby największą ilość gotowych talii tak szybko jak to możliwe.

```
MATCH (u:User {Name: 'Ala Warzyńska'})
MATCH (d:Deck)<−[:IN_DECK]−(c:Card)
OPTIONAL MATCH (u)−[o:OWNS]−>(c)
WITH d, c, COUNT(o) AS owns_card
WHERE owns_card = 0
WITH d, COLLECT(c) AS missing_cards, COUNT(c) AS missing_cards_count
ORDER BY missing_cards_count ASC
UNWIND missing_cards AS mc
RETURN mc.Name AS card_name, mc.Price as card_price, d.Name AS deck
```

card_name	card_price	deck
"Blue-Eyes Shining Dragon"	52.38	"Blue-Eyes Evolution"
"Blue-Eyes Shining Dragon"	52.38	"Blue-Eyes Starter"
"Eternal Soul"	27.82	"Magician's Force"

5. Algorytmy grafowe - Ważność wierzchołków Degree

Tworzenie grafu:

```
CALL gds.graph.project('graf',
    ['Card','User','Type','Deck'],
    {
        FRIENDS:{orientation:'UNDIRECTED'},
        OWNS:{orientation:'REVERSE'},
        TYPE:{orientation:'REVERSE'},
        IN_DECK:{orientation:'REVERSE'}
    }
);
```

Zastosowanie algorytmu:

```
CALL gds.degree.stream("graf") YIELD nodeId,score
RETURN
CASE WHEN gds.util.asNode(nodeId).Name IS NOT NULL THEN gds.util.asNode(nodeId
    ).Name
ELSE gds.util.asNode(nodeId).Type
END as nodeName,
score
ORDER BY score DESC;
```

Wynik:

nodeName	score
"Spell"	19.0
"Phantom's Madness"	15.0
"Monster"	14.0
"Dark Magician Chronicles"	12.0
"Magician's Force"	11.0
"Chaos Masters"	10.0
"Kaiba Duelist Deck"	10.0
"Aphermage"	9.0

6. Algorytmy grafowe - Ważność wierzchołków PageRank

Tworzenie grafu:

```
CALL gds.graph.project('pageGraf',
    ['Card','User','Type','Deck'],
    {
        FRIENDS:{orientation:'UNDIRECTED'},
        OWNS:{orientation:'NATURAL'},
        TYPE:{orientation:'NATURAL'},
        IN_DECK:{orientation:'NATURAL'}
    }
);
```

Zastosowanie algorytmu:

```
CALL gds.degree.stream("graf") YIELD nodeId,score
RETURN
CASE WHEN gds.util.asNode(nodeId).Name IS NOT NULL THEN gds.util.asNode(nodeId
    ).Name
ELSE gds.util.asNode(nodeId).Type
END as nodeName,
score
ORDER BY score DESC;
```

Wynik:

nodeName	score
"Spell"	1.0464066027052235
"Phantom's Madness"	0.7728593204205417
"Monster"	0.7490218220427372
"Dark Magician Chronicles"	0.697853335915435
"Magician's Force"	0.6041603807179703
"Chaos Masters"	0.5546192643317551
"Aphermage"	0.5483072994906468
"Kaiba Duelist Deck"	0.5465627535446457

6. Algorytmy grafowe - Beta Closeness

Tworzenie grafu:

```
CALL gds.graph.project('close',
    ['Card','User','Type','Deck'],
    {
        FRIENDS:{orientation:'UNDIRECTED'},
        OWNS:{orientation:'UNDIRECTED'},
        TYPE:{orientation:'UNDIRECTED'},
        IN_DECK:{orientation:'UNDIRECTED'}
    }
);
```

Zastosowanie algorytmu:

```
CALL gds.beta.closeness.stream("close") YIELD nodeId,score
RETURN
CASE WHEN gds.util.asNode(nodeId).Name IS NOT NULL THEN gds.util.asNode(nodeId
    ).Name
ELSE gds.util.asNode(nodeId).Type
END as nodeName,
score
ORDER BY score DESC;
```

Wynik:

nodeName	score
"Ala Warzyńska"	0.7088607594936709
"Łukasz Kucki"	0.5544554455445545
"Pot of Greed"	0.5233644859813084
"Blue-Eyes Chaos MAX Dragon"	0.49122807017543857
"Dragon Master Knight"	0.4827586206896552
"Ultimate Fusion"	0.47863247863247865
"Heavy Storm"	0.47863247863247865
"Spell"	0.4745762711864407
"Blue-Eyes Alternative White Dragon"	0.4745762711864407
"The Eye of Timaeus"	0.4745762711864407