EXERCISE 1

Consider the database:

TEAMS *(team_id, team_name, city, nation)*

PLAYERS *(player_id, name, surname, position, team, nationality)*

MATCHES *(match_id, date, home_score, away_score, home_team, away_team)*

GOALS *(goal_id, match, player, goal_time)*

The stored data includes the championships in Europe for Italy, Spain, France, and England. The federation wants to split the database so that each nation manages the matches of its own championship. Note that foreign players can play in foreign teams.

ESERCIZIO 1

PRIMA LA CHIAVE PRIMARIA E I COLLEGAMENTI ALLE CHIAVI ESTERNE

TEAMS (TEAM_ID, TEAM_NAME, CITY, NATION)

PLAYERS (PLAYER_ID, NAME, SURNAME, POSITION, TEAM, NATIONALITY)

MATCHES (MATCH_ID, DATE, HOME-SCORE, AWAY_SCORE, HOME_TEAM, AWAY TEAM)

GOALS (GOAL_ID, MATCH, PLAYER, GOAL_TIME)

INIZIAMO A DIVIDERE LE MACRO TEAM VISTO CHE C'È A SE STANTE

$$TEAM\_ITALY = \sigma_{NATION = "ITALY"}(TEAMS)$$

$$TEAM\_SPAIN = \sigma_{NATION = "SPAIN"}(TEAMS)$$

SELECT USARE

FILTRO SU NATION, COME DA CONSEGNA

" _ NAME
" _ SURNAME

UGUALI

PROIEZIONE → SELECT

VISTO CHE CE L'ABBIAMO QUA

$$PLAYERS\_ITALY = \Pi_{PLAYER\_ID, NAME, SURNAME, \dots}(PLAYERS \bowtie_{TEAM = TEAM\_ID} TEAM\_ITALY)$$

// LE ALTRE SONO UGUALI

$$MATCHES\_ITALY = \Pi_{MATCH\_ID, DATE, HOME\_SCORE, \dots}(MATCHES \bowtie_{HOME\_TEAM = TEAM\_ID \text{ AND } AWAY\_TEAM = TEAM\_ID} (\text{PRIMO MANCHI } TEAM\_ITALY))$$

FORSE NON CAPISCO ASSUNZIONE CHE NON C'È SCUSA

IN QUESTO CASO L'OR NON VANNE

ANDREBBERO PRESE 2 SOLE SQUADRE IN REALTÀ

BASTA SCRIVERLO IN ITALIANO NEL CASO

$$GOALS\_ITALY = \Pi_{GOAL\_ID, MATCH, \dots}(GOALS \bowtie_{PLAYER = PLAYER\_ID} PLAYERS\_ITALY)$$

EXERCISE 2

Express the following queries on transparency levels of fragmentation and allocation transparency (referred to the previous exercise):

- → *Extract all the match results of the "Milan" (Italy) teams*
- → *Extract the number of midfielders (position) in france*
- → *Determine the number of goals scored after the minutes 75' in all championships (return a single value that represents the sum of all championships)*
- → *Determine the number of goals scored in Italy for each minute (from 1 to 90) → Soccer matches last 90 minutes*

Esercizio 2

1 FRAGMENTATION, ALLOCATION

```
SELECT HOME_SCORE, AWAY_SCORE
FROM MATCHES_ITALY AS M
JOIN TEAMS_ITALY AS T
ON M.HOME_TEAM = T.TEAM_ID
        OR  → NON CI INTERESSA SE È IN MATCHES ON CASA
    M.AWAY_TEAM = T.TEAM_ID
WHERE city = "MILAN"
```

ANCHE CHE SAPPIAMO CHE I TEAM DI MILANO SONO IN ITALIA

2 FRAGM:

```
SELECT COUNT(*) AS MIDF_FRANCE
FROM PLAYERS AS P
JOIN TEAMS AS T
ON P.TEAM = T.TEAM_ID
WHERE T.NAME = "FRANCE" AND P.POSITION = "MIDFIELD"
```

ALLOCATION:

```
SELECT COUNT(*) AS MDF_FRANCE
FROM PLAYERS_FRANCE
WHERE POSITION = "MIDFIELD"
```

3 FRAGMENTATION:

```
SELECT COUNT(*)
FROM GOALS
WHERE GOAL_TIME > 75
```

ALLOCATION → RELAZIONE TEMPORANEA

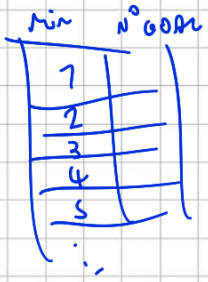
```
CREATE VIEW 75_GOAL AS
SELECT *
FROM GOALS_ITALY
WHERE GOAL_TIME > 75
UNION
  SELECT *
  FROM GOALS_SPAIN
  WHERE GOAL_TIME > 75
```

UNION
⋮

SELECT COUNT(*)
FROM FS_GOALS

4 FRAMMENTAZIONE ORIZZONTALE:

SELECT COUNT(*), GOAL_TIME
FROM GOALS AS G
JOIN MATCH AS M
ON G.MATCH = M.MATCH_ID
JOIN TEAMS AS T
ON T.TEAM_ID = M.HOME_TEAM OR
T.TEAM_ID = M.AWAY_TEAM
WHERE NATION = "ITALY"
GROUP BY GOAL_TIME



SBAGLIATO, BISOGNA FARE UN ALTRO JOIN

JOIN TEAMS AS T2
ON T2.TEAM_ID = M.AWAY_TEAM

WHERE T.NATION = "ITALY" AND
T2.NATION = "ITALY"

↓

LA QUERY VUOLE TUTTI GOAL
: TEAM CASA ITALIANI

↓

L'OMOGENEITÀ INIZIALE DEI
DATI I TEAM POSSONO NON ESSERE
CON TEAM DI NAZIONI DIVERSE

ALLOCAZIONE TRASPARENTE:

SELECT COUNT(*), GOAL_TIME
FROM GOAL_ITALY
GROUP BY GOAL_TIME

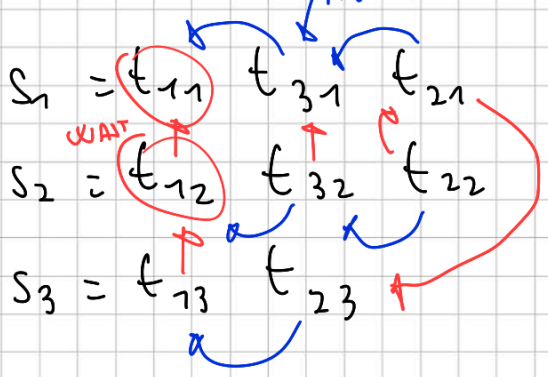
Esercizio 3

$t_1 = \underbrace{r_{11}(x)\, w_{11}(x)}_{t_{11}} \underbrace{\| r_{12}(y)\, w_{12}(y) \|}_{t_{12}}$

NON CI SCRIVO TUTTO

NON ASPETTARE CHE FINISCA LA PRIMA

PROCESSO 3 CHE OPERA SUL NODO 1



* REMOTE PROCEDURE CALLS → PASSO DI AVERLI SCRITTI TUTTI

NO CICLI, NO DEADLOCK

EXERCISE 3

Given three distributed transactions *t1*, *t2* and *t3* operating on resources *X* and *Y*:

t1 → r11(X)w11(X)r12(Y)w12(Y)r13(Z)w13(Z)

t2 → r23(Z)w23(Z)r21(X)w21(X)r22(Y)w22(Y)

t3 → r31(X)w31(X)r32(Y)w32(Y)

Let's assume that the three transactions execute on nodes 1, 2 and 3, presenting the following schedules:

S1 → r11(X)w11(X)r31(X)w31(X)r21(X)w21(X)

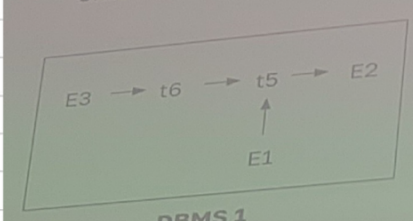
S2 → r12(Y)w12(Y)r32(Y)w32(Y)r22(Y)w22(Y)

S3 → r13(Z)w13(Z)r23(Z)w23(Z)

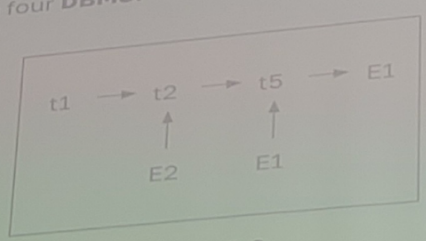
Is it a **deadlock** situation?

EXERCISE 4

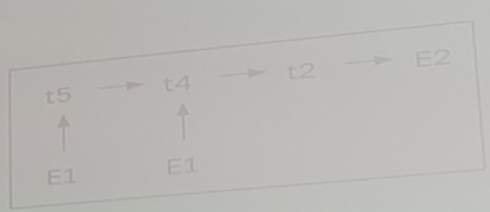
Given the following situation on four **DBMS**:



DBMS 1



DBMS 2



DBMS 3

Simulate the deadlock detection algorithm step by step:

Esempio 4

SEQUENZE DI ATTESA CRITICHE: → ESISTE UN'ATTESA, BISOGNA CAPIRE DOVE PROPAGARLA

$E_k \to t_i \to t_j \to E_l$

E_k → NODO; t_j → ARCHI; E_l → NODO DOVE L'ATTESA SI PROPAGA

$t_i \to t_j$ t.c. $i > j$

t_i → CONDIZIONE PER PROPAGARE L'ATTESA

LA SEQUENZA VA PROPAGATA AL NODO l

DBMS 1 SCRIVIAMO SOLO LE SEQUENZE CHE SI POSSONO PROPAGARE

$E_3 \to t_6 \to t_5 \to E_2$

E_3 ATTENDE t_6; t_6 ATTENDE t_5; CONDIZIONE SU E_2 ATTESA CHE t_5 TERMINI

PROPAGHIAMO L'ATTESA SU l, OVVERO IL NODO 2

NULLA SU DBMS 2

DBMS 3

$E_1 \to t_5 \to t_4 \to t_2 \to E_2$

NODO 1 NODO 3

NODO 2:

t_4 t_6

$t_1 \to E_2 \to t_5 \to E_1$

E_2 E_2

DEADLOCK C'È UN CICLO

ESERCIZIO 5

UNA QUERY CON FIND E UNA CON AGGREGATION PIPELINE

1

SELEZIONO, PRIMO PARAMETRO (CRITERIO)

```
DB.BOOKS.FIND({
    "PUBLISHED_YEAR": {"$GT": 1850},
    "AVAILABLE": TRUE
},
```

SECONDO PARAMETRO, PROIEZIONE (SELECT)

É: MOSTRI TUTTO INCLUSO ANCHE

↓

BISOGNA ESCLUDERLO

↓

```
{"TITLE": 1, "AUTHOR.NAME": 1, "AUTHOR.SURNAME": 1, "GENRE": 1, "_ID": 0}
)
```

2 NON VOGLIO RICHIESTA SELEZIONE QUINDI PRIMA È VUOTA

```
DB.BOOKS.AGGREGATE([{
    "$GROUP": {"_ID": "$GENRE",
               "TOTAL_PAGES": {"$SUM": "$TOTAL_PAGES"}
}}])
```

CI VUOLE _ID, ALTRIMENTI NON SA COME RAGGRUPPARE

EXERCISE 5

Given the following collection of "books", an example document of which is provided, write the queries listed below using the syntax covered in class

```
{
    "_id": ObjectId("60d5ecf8d6e2f5a5d8f1e9a2"),
    "title": "Il nome della rosa", "Surname": "Eco" },
    "author": { "name": "Umberto",
    "genre": "novel",
    "total_pages": 533
    "published_year": 1980,
    "available": false,
    "borrowed by": { "name": "Alice", "Surname": "Rossi" },
    "borrow_date": ISODate("2024-05-01T10:00:00Z")
}
```

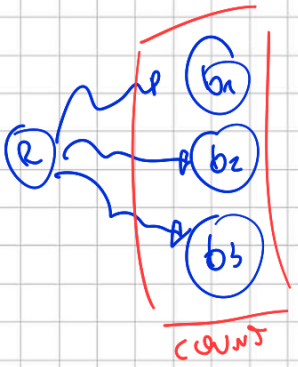
→ Find all the books published after the 1850 that are available in the bookshelf. Project only the *title*, author *name*, *surname* and the *genre*

→ Using the aggregation pipeline, find the number of total pages considering each book of each genre

Esercizio 6

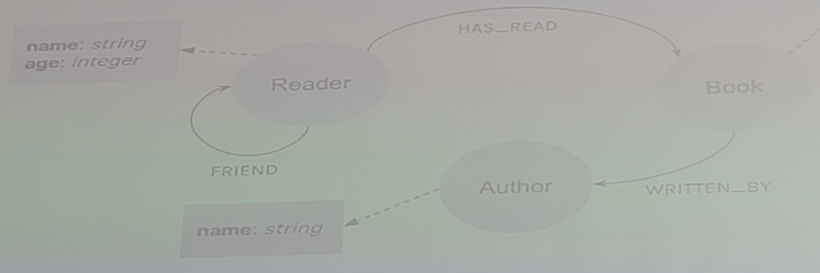
1
```
MATCH (r:Reader)-[:HAS_READ]->(b:Book)-[:WRITTEN_BY]->(a:Author)
WHERE r.NAME = "Elda Scialtiel"
RETURN r, b, a
```

2
```
MATCH (r:Reader)-[:HAS_READ]->(b:Book)
WITH r, count(b) AS NUMBER_OF_BOOKS
WHERE b.PUBLISHED_YEAR > 1930
RETURN r.NAME, NUMBER_OF_BOOKS
```



EXERCISE 6

Given the following collection of "books", an example document of which is provided, write the queries listed below using the syntax covered in class



- Find all the books read by "Elda Scialtiel" and the respective author. Return the nodes of the extracted paths
- Find how many books each reader has read with a publication date after 1930. Return the name of the reader and the number of read books

Esercizio 7 — inverted index

ID	TOKEN	DOCS
1	LLMs	1,2,3
2	data	1,2
3	NLP	3
4	AI	3
5	text	1,3

EXERCISE 7

Given the following 3 documents:

DOC 1: LLMs are able of generating human-like text based on the data they receive

DOC 2: Training LLMs require significant resources and vast amounts of data

DOC 3: The applications of LLMs span various fields, including text in NLP and AI

Build the “Apache Lucene” inverted index for these documents

EXERCISE 8

Given an index with 5 documents, compute the "Apache Lucene" scores (such as Elasticsearch) when searching the text: "*LLMs in AI*" by knowing that the following 3 documents are extracted (suppose to cut off stopwords such as "in" in the searching text):

DOC 1: *LLMs are able of generating human-like text based on the data they receive*

DOC 2: *Training LLMs require significant resources and vast amounts of data*

DOC 3: *The applications of LLMs span various fields, including text in NLP and AI*

Esercizio 8

TERM FREQUENCY – INVERSE DOCUMENT FREQUENCY

FORMULA $TF\text{-}IDF(t,d) = TF(t,d) \cdot IDF(t)$

TOKEN, DOCUMENTO

DA APPLICARE PER OGNI DOCUMENTO

TERM FREQUENCY → NUMERO DI VOLTE CHE t APPARE NEL DOCUMENTO

IDF INDEX → QUANTO È RARO IL TERMINE TRA TUTTI I DOCUMENTI

TEXT = "LLMs AI"

$IDF("LLMs") = \log\left(\frac{5}{3}\right) = 0,222$

$IDF("AI") = \log\left(\frac{5}{1}\right) = 0,699$

$IDF = \log\left(\frac{n}{df(d,t)}\right)$

NUMERO TOTALE DI DOCUMENTI NELL'INDICE

QUANTE VOLTE È PRESENTE NEI DOCUMENTI

$TF\text{-}IDF("LLMs", doc\_1) = 1 \cdot 0,222 = 0,222$

$TF\text{-}IDF("AI", doc\_1) = 0$

$TF\text{-}IDF("LLMs", doc\_2) = 1 \cdot 0,222 = 0,222$

$TF\text{-}IDF("AI", doc\_2) = 0$

$TF\text{-}IDF("LLMs", doc\_3) = 1 \cdot 0,222 = 0,222$

$TF\text{-}IDF("AI", doc\_3) = 1 \cdot 0,699 = 0,699$

$Score(d) = \sum_{t} TF\text{-}IDF(t,d) \cdot Norm(d)$

CLASSIC SEARCH ASSUME CHE UN DOCUMENTO CHE HA PIÙ TOKEN VALE DI MENO RISPETTO A UNO PIÙ CORTO → QUINDI NORMALIZZA RISPETTO ALLA LUNGHEZZA DEL DOCUMENTO

$Norm(d) = \frac{1}{\sqrt{size(d)}}$

$Norm(doc\_1) = \frac{1}{\sqrt{13}} = 0,277$

$Norm(doc\_2) = 0,316$

$Norm(doc\_3) = 0,277$

$Score(doc\_1) = 0,222 \cdot 0,277 = 0,061$

$Score(doc\_2) = 0,222 \cdot 0,316 = 0,070$

$Score(doc\_3) = (0,222 + 0,699) \cdot 0,277 = 0,255$

Esercizio 9

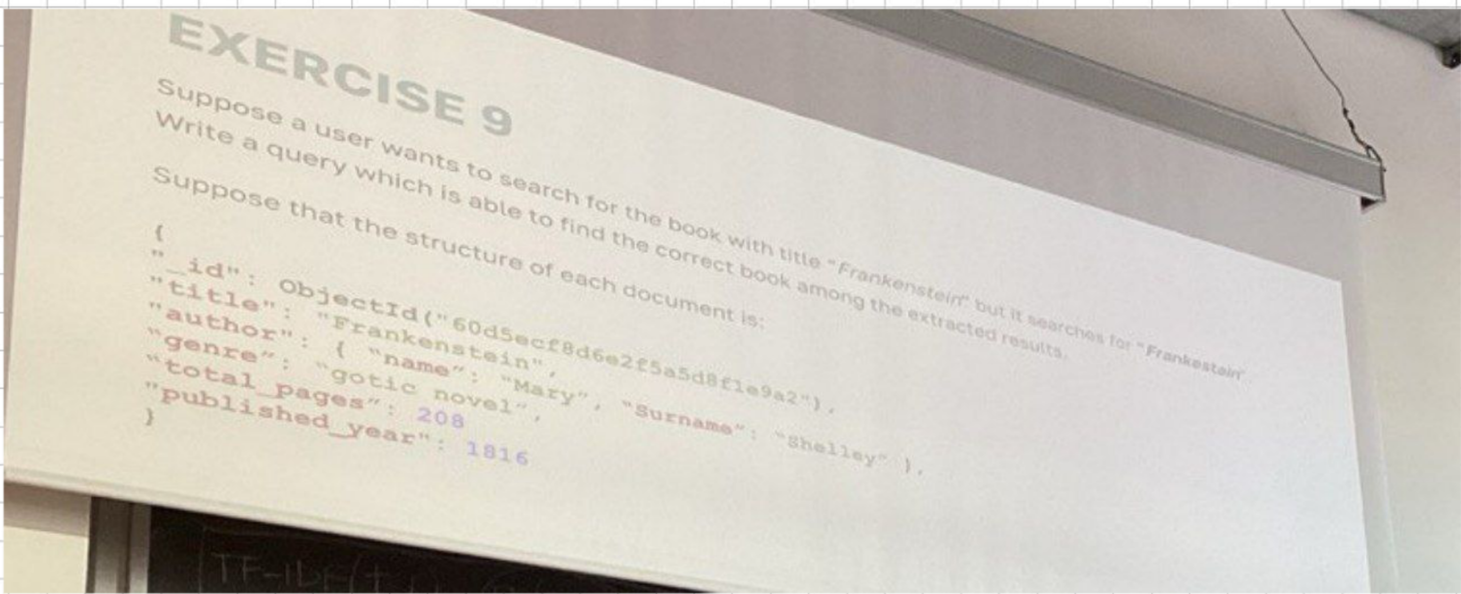
Query Elasticsearch

```
{"query": {
    "bool": {
        "should": [
            "match": {"title": {"query": "Frankestain",
                                "fuzziness": "auto"}}
            }
        ]
    }
}}
```

Usiamo le boolean queries, accanto un parametro should o must in questo caso non cambia

Frankenstein
Frankestain > GD = 2 → fuzziness va bene perché funziona fino a 2

fuzziness → Aggiungiamo ngrams se la GD è > 2 → "title.ngrams" → divide la query in tokens per trovarne di simili



Esercizio 10

SCHEMA MATCHING:

BISOGNA GIUSTIFICARE

ISBN = ISBN → PERCHÉ MATCH ESATTO

TITLE = BOOK TITLE → MATCH SUBSTRING

AUTHOR = BOOK AUTHOR → MATCH SUBSTRING

PRICE = PRICE → MATCH ESATTO

UNIONE TABELLE

UT (ISBN, TITLE, AUTHOR, PRICE)

- RIGA 1 DELLA TABELLA A → RIGA 1 DELLA TABELLA B
  - ISBN COINCIDONO
- RIGA 2 DELLA TABELLA A → VIENE AGGIUNTA
- RIGA 2 DELLA TABELLA B → NON È CONSISTENTE (PER COLPA DELL'ISBN NON CORRETTO)

SCEGLIAMO DI MODIFICARE LA TABELLA PIÙ AGGIORNATA

ISBN	TITLE	AUTHOR	PRICE
02$----	MOBY DICK	MELVILLE	20$
99$----	1984	ORWELL	30$

← IL PREZZO CORRETTO È 30$ PERCHÉ LA LIBRERIA CHE ACQUISISCE L'ORDINE

EXERCISE 10

Two libraries A and B must merge their book databases. Suppose that library A acquired library B and note that ISBN number must have 13 characters. Your task is to perform the integration of "Book" tables that are available below:

Table A → Last Update: **04-2024**

ISBN	title	author	price
9781503280786	Moby Dick	Melville	20$
9780451524935	1984	Orwell	30$

Table B → Last Update: **10-2022**

ISBN	book title	book author	price
9781503280786	Moby Dick	Melvle	15$
97804524935	1984	Orwel	45$