# Group Project - Neo4j

of Systems and Methods for Big and Unstructured Data Course (SMBUD)

held by

Brambilla Marco

Tocchetti Andrea

## Group 78


Pisante Giuseppe 10696936

Raffaelli Martina 10709893


Academic year 2024/2025

POLITECNICO
MILANO 1863

# Contents

# 1 Introduction

The project aims to design and implement a database system to support the management of data related to the film industry. The database will include entities such as Person (actor, director, writer), Title (film, TV series), Episode, ratings, and genre. The goal is to create a comprehensive system that can store and query information about films, TV series, and the people involved in their production. The project will develop with Neo4j, to exploit the relations between the entities and find significant insights from the data, such as collaborations between directors and actors, trend of genres over time, most working actors and the most successful films.

# 2 Assumptions

The project is based on the following assumptions:

- Each person has a unique ID and a name, surname, and date of birth. Some could also have a death date, if passed away.
- Each person can be associated with multiple roles (actor, director, writer, archive footage, music department, producer)
- Each title has a unique identifier and a title type (film, TV series, shortfilm)
- Each title can have multiple episodes
- Each title can have multiple genres
- Each user can rate a title if and only if it has watched the film
- Each title has a unique rating, average rating from the users, and the number of votes
- Each person can be associated with multiple titles
- Each title can have multiple people associated with it

# 3 ER diagram


**Figure 1:** E-R Diagram

## 3.1 Entities

Starting from the considerations previously exposed regarding the implementation hypotheses, we have drawn an ER diagram (**Figure 1**) which includes 5 different entities and 7 many-to-many relationships described below in the logical model:

- **Person**(nconst, PrimaryName, BirthYear, DeathYear, PrimaryProfession, KnownForTitles)
- **Title**(tconst, PrimaryTitle, OriginalTitle, TitleType, StartYear, EndYear, RuntimeMinutes, Genres)
- **Episode**(tconst, ParentTconst, SeasonNumber, EpisodeNumber)
- **Ratings**(tconst, AverageRating, NumVotes)

- **Genre**(Name)

The **Person** entity describes every possible individual with their own personal data, including their primary profession and titles they are known for. The **Title** entity represents films, TV series or short films with their respective attributes. The **Episode** entity is used to detail episodes of TV series, linked to their parent series. The **Ratings** entity captures the average rating and number of votes for each title. Finally, the **Genre** entity categorizes the titles into different genres.

### 3.2 Relationships

**ACTED_IN** $(Person) - [: ACTED\_IN] -> (Title)$
Relationship between a Person, whose primary profession is actor, and a Title.

**DIRECTED** $(Person) - [: DIRECTED] -> (Title)$
Relationship between a Person, whose primary profession is director, and a Title.

**WROTE** $(Person) - [: WROTE] -> (Title)$
Relationship between a Person, whose primary profession is writer, and a Title.

**PART_OF** $(Episode) - [: PART\_OF] -> (Title)$
Relationship between an Episode and its parent Title, whose TitleType is TV series.

**HAS_GENRE** $(Title) - [: HAS\_GENRE] -> (Genre)$
Relationship between a Title and a Genre.

**HAS_RATING** $(Title) - [: HAS\_RATING] -> (Rating)$
Relationship between a Title and its Rating.

### 3.3 Constraints:

ciao

## 4 Cypher Queries

### 4.1 DistributionCenter closest to each User

```
MATCH (u:User), (d:DistributionCenter)
WITH u, d, distance(point({latitude: u.latitude, longitude: u.longitude}),
                    point({latitude: d.latitude, longitude: d.longitude})) AS
RETURN u.id AS user_id, d.name AS nearest_center, dist
```

```
ORDER BY dist ASC
LIMIT 10;
```

| user_id | nearest_center | dist |
|---|---|---|
| 58210 | Chicago IL | 1759.9113 |
| 48442 | Chicago IL | 1759.9113 |
| 20571 | Chicago IL | 1759.9113 |
| 37767 | Philadelphia PA | 1897.4164 |
| 7543 | Philadelphia PA | 1897.4164 |
| 41572 | Philadelphia PA | 1897.4164 |

**Table 1:** Output della query Neo4j.

## 4.2 Most sold products:

```
MATCH (:OrderItem)-[:REFERS_TO]->(p:Product)
RETURN p.name AS product_name, count(*) AS sales_count
ORDER BY sales_count DESC
LIMIT 10;
```

| Product Name | Quantity |
|---|---|
| Wrangler Men's Premium Performance Cowboy Cut Jean | 62 |
| Puma Men's Socks | 48 |
| 7 For All Mankind Men's Standard Classic Straight Leg Jean | 41 |
| True Religion Men's Ricky Straight Jean | 37 |
| Kenneth Cole Men's Straight Leg Jean | 36 |
| Michael Kors Men's 3 Pack Brief | 33 |
| HUGO BOSS Men's Long Pant | 31 |
| Lucky Brand Mens Men's 361 Vintage Straight Denim Jean | 31 |
| Thorlo Unisex Experia Running Sock | 31 |
| Diesel Men's Shioner Skinny Straight Leg Jean | 31 |

**Table 2:** Top-selling men's clothing products and their quantities.

# 5 References & Sources


[1] Course Slides

[2] https://pysimplegui.readthedocs.io/en/latest/call

[3] https://py2neo.org/

[4] https://neo4j.com/docs/cypher-manual/current/

[5] https://neo4j.com/developer/python/

[6] http://iniball.altervista.org/Software/ProgER

[7] https://neo4j.com/developer/cypher/

[8] https://pandas.pydata.org/docs/