Postgres

* MovieCredits(id\_movie,id\_credit)
* Credits(id\_credit,id\_cast,id\_actor,character)
* Actors(id\_actor,name,gender,profile\_path,popularity,birthday)
* TVRoles(id\_actor,id\_credit\_Tv,id\_serie,character,episode\_count)
* TvShow(id\_serie,name,original\_name,seasons\_number,episodes\_number,status,original\_language,vote\_average,popularity,poster\_path)

Mongo DB

* Movies(id\_movie, movie\_json)
* MovieKeywords(id\_movie, keywords\_json)

Neo4j

* Movie (id\_movie)
* Review (id\_review,id\_movie, content)
* User(nick)
* Movie ->[:hasReview]->Review
* Review ->[:writtenBy]->User
* Movie->[:isRelated]->Movie

Cassandra

* MovieTranslations (id\_movie, translation\_languages, translation\_Json )
* MovieAlternativeTitles (id\_movie, iso\_3166 , alternative\_titles)
* ActorDailyRanking (timestamp, rank, popularity\_score, id\_actor, name, known\_for\_json)

Giallo = lista in Cassandra

Rank = {1,…,N}

iso\_3166\_1

iso\_639\_1

Redis

* MovieTrailer:id\_Movie lista di trailer
* MoviePosters:id\_Movie lista di immagini
* PeopleProfiles:id\_people lista di immagini // ma gli id degli attori li prendiamo da quelli presenti in Postgres dove ci sono i credits dei film già inseriti ?

Idee per i trailer: un’alternativa è avere un hash per ogni movieTrailer (o per ogni moviePoster) dove all’interno si memorizzano i relativi trailer (o le relative immagini) con una scelta di chiave con nome incrementale i.e. trailer1:trailer\_url, trailer2:trailer\_url

Per i poster un’alternativa poteva essere avere una lista di stringhe cosi formattata “(score)http://urlImg” al fine di memorizzarsi anche l’informazione dello score dell’immagine. Oppure dovremmo memorizzare le prime n immagini ordinate per score?