

Álvaro de la Flor Bonilla





# CONFIGURACIÓN



#### LENGUAJE



#### BASE DE DATOS





# **DATASET**

#### MOVIELENS 100K DATASET

#### 100K Dataset

ings. Stable benchmark datas

MB, checksum)

S

is.org/datasets/movielens/100



# Desarrollo

#### MONGODB, RECOMENDADOR

e Header<th sty



#### Lectura y guardado

```
def read_u_user():
    with open(get_url_u_user()) as f:
        lines = f.readlines()
        for line in lines:
           user_id = x[0].strip()
            age = x[1].strip()
           gender = x[2].strip()
           occupation = x[3].strip()
            zip\_code = x[4].strip()
           user = User(user_id=user_id, age=age, gender=gender, occupation=occupation, zip_code=zip_code)
            users.append(user)
    return users
def save_users(rates):
    db = get_mongo_db()
    users_mongodb = db.users
    users_mongodb.drop()
    user_dict = []
    for rate in rates:
        user_dict.append(json.loads(rate.to_json()))
    users_mongodb.insert_many(user_dict)
```

#### giw\_db.movies

COLLECTION SIZE: 412.85KB TOTAL DOCUMENTS: 1682 INDEXES TOTAL SIZE: 32KB

Find

movie id: "1"

release date: "1995-01-01 00:00:00"

title: "Toy Story (1995)" video\_release\_date: ""

Indexes

Schema Anti-Patterns

Aggregation

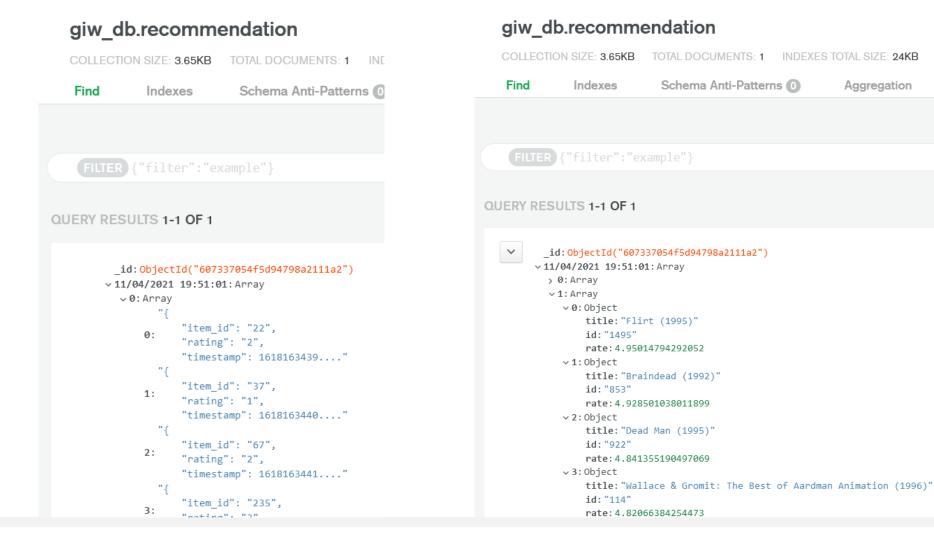
```
FILTER {"filter":"example"}
QUERY RESULTS 1-20 OF MANY
          _id: ObjectId("607086cbeb38f9a28fe2354e")
        ∨ genres: Array
            0: "Animation"
            1: "Children s"
            2: "Comedy"
         imdb_url: "http://us.imdb.com/M/title-exact?Toy%20Story%20(1995)"
```

```
_id: ObjectId("607086cbeb38f9a28fe2354f")

√ genres: Array

   0: "Action"
   1: "Adventure"
   2: "Thriller"
 imdb url: "http://us.imdb.com/M/title-exact?GoldenEye%20(1995)"
 movie id: "2"
```

### Lectura y guardado



Aggregation

#### Recomendación

```
from math import sqrt
 from utilities.rate_util import *
 from utilities.constants import *
 import shelve
def load_dict(rates_add):...
def get_pearson(dict_user, user_1, user_2):...
def top_matches(dict_user, person, similarity=get_pearson):...
def get_recommendations(dict_user, person, similarity=get_pearson):...
def transform_dict_user(dict_user):...
```



# Desarrollo

#### **DEMO**

