# MongoDB

Gabriel Rodríguez Flores

February 13, 2024

ADA ITS
INSTITUTO TECNOLÓGICO SUPERIOR

- Aggregation

## Contents

# 1 Teoría

- MongoDB Playground

## 1.1 Aggregate

### 1.1.1 Sintaxis

```
db.getCollection('movies').aggregate([
  { /* First stage of pipeline */ },
  { /* Second stage data come from first stage */ },
  ...
  { /* Last stage of pipeline */ },
])
```

### 1.1.2 Operaciones

- Lista de operadores

### 1.1.3 Stages

- **Lista de etapas**

Se listan los más utilizados.

> Ejemplos sobre la base de datos *sample_mflix* de Mongodb Atlas

- Se pueden usar los ya conoceidos: `$sort`, `$skip`, `$limit`, `$count`

```
db.getCollection('movies').aggregate([
    { $sort : { _id : -1 } },
    { $skip : 50 },
    { $limit : 50 },
    { $count : 'total' },
])
```

- $match Equivalente a los filtros de find

```
db.getCollection('movies').aggregate([
  { $match: {
      year: { $gte: 1993 }
  }},
])
```

- $group Agrupar por un parámetro

```
db.getCollection('movies').aggregate([
    { $group: {
        _id: '$year',
        first: { $first: '$$ROOT' },
        last: { $last: '$$ROOT' },
        all: { $push: '$$ROOT' },
        total: { $count: {} },
        imdbAverage: { $avg: '$imdb.rating' }
    }}
])
```

- $addFields Añade un campo nuevo

```
db.getCollection('movies').aggregate([
    { $addFields: {
        rating:  {
            $avg: [
                "$imdb.rating",
                { $multiply: [2, "$tomatoes.viewer.rating"] },
                "$tomatoes.critic.rating"
            ]
        }
    }}
])
```

- $lookup Rellena (Similar a Join de SQL)

```
db.getCollection('movies').aggregate([
    { $lookup: {
        from: 'comments',
        localField: '_id',
        foreignField: 'movie_id',
        as: 'comments'
    }}
])
```

- $unwind Separar un array en varios documentos con valor único por cada elemento del array

```
db.getCollection('movies').aggregate([
    { $unwind: '$genres'}
])
```

- $project Para seleccionar los datos que se quieren incluir o quitar

```
db.getCollection('movies').aggregate([
    { $project: {
        _id: 0,
```

```
        title: 1,
        genres: 1,
        year: 1,
        poster: 1
    }}
])
```

- $facet Lanzar diferentes pipeline de una sola vez

```
db.getCollection('movies').aggregate([
    { $facet: {
        moviesSortedByYear: [
            { $sort : { year : -1 } },
            { $project: {
                _id: 0,
                title: 1,
                year: 1,
            }},
        ],
        moviesSortedByImdbRating: [
            { $sort : { 'imdb.rating' : -1 } },
            { $project: {
                _id: 0,
                title: 1,
                'imdb.rating': 1,
            }},
        ],
    }}
])
```

### 1.2 Avanzados

- Optimización del uso de aggregate

## 2 Ejemplos

- Documentación oficial

## 3 Ejercicios

1. Devolver la cuenta de cuántas películas y series hay en español
    - Añadir los nombres de cada película en un array

- En lugar de array de nombres, que sean objetos que contengan:
  - nombre
  - año
  - valoración imdb
  - géneros

2. Devolver las películas agrupadas:
   - Primer nivel por género
   - Segundo nivel por año
3. Agrupar las películas por valoración
   - La agrupación se hace por valores enteros. Ej: en la categoría 7 estarán todos los comprendidos entre `[7, 8)`
4. Agrupar las películas por categorias y coger sólo
   - nombre
   - año
   - valoración media calculada

# 4 Entregables

## 4.1 En clase

## 4.2 Tarea