Sistemas de Información para el Análisis de GVDatos

Instituto de Computación - Facultad de Ingeniería

Mayo 2024

BASES DE DATOS NOSQL

Contenido

- Bases de Datos de Documentos
  - JSON
- Bases de Datos Columnares
  - Cassandra
- Bases de Datos de Grafos
  - Neo4J

Introducción

- Modelo visto en el curso:
  - Bases de Datos Relacionales (BDR)
- Existen otros modelos de datos:
  - Bases de datos documentales
  - Bases de datos columnares
  - Bases de datos de grafos

Bases de Datos de Documentos

- BD NO relacional que almacena datos como documentos estructurados.
  - Normalmente en formatos JSON.

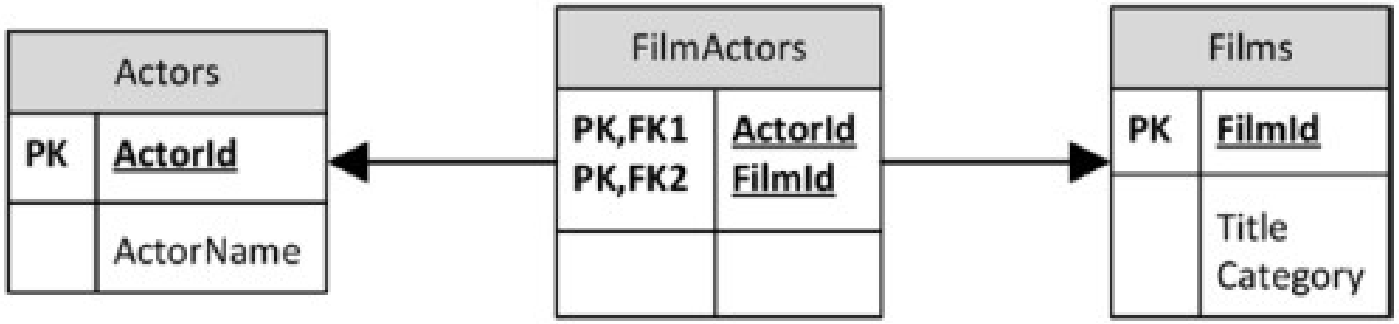
- Formato JSON
  - XML es el punto de partida para el formato JSON.
  - Es un formato de intercambio de datos.
  - Apunta a soportar aplicaciones web operacionales.

BD de Documentos: JSON

- La jerarquía de almacenamiento suele ser la siguiente:
  - un documento es la unidad básica de almacenamiento
  - un documento comprende uno o más pares clave-valor (key-value)
  - puede contener documentos anidados y matrices (arrays).
  - Los arrays también pueden contener documentos que permitan una estructura jerárquica compleja.
  - Una colección es un conjunto de documentos que comparten un propósito común.
  - Los documentos de una colección no tienen que ser del mismo tipo, aunque es común que representen cierta información.

BD de Documentos: JSON

- La siguiente es una BDR de películas y actores:



- En una BD de documentos JSON:

```
{ "_id" : 97, "Title" : "BRIDE INTRIGUE",
              "Category" : "Action",
    "Actors" :
      [ { "actorId" : 65, "Name" : "ANGELA HUDSON" } ]
}

{ "_id": 115,"Title": "CAMPUS REMEMBER",
             "Category": "Action",
    "Actors" :
       [
         { "actorId": 8,"Name": "MATTHEW JOHANSSON" },
         { "actorId": 45,"Name": "REESE KILMER" },
         { "actorId": 168,"Name": "WILL WILSON" }
       ]
}

{ "_id" : 105, "Title" : "BULL SHAWSHANK",
               "Category" : "Action",
     "Actors" :
       [ { "actorId" : 2, "Name" : "NICK WAHLBERG" },
         { "actorId" : 23, "Name" : "SANDRA KILMER" } ]
}
```

Actor document

Film Document

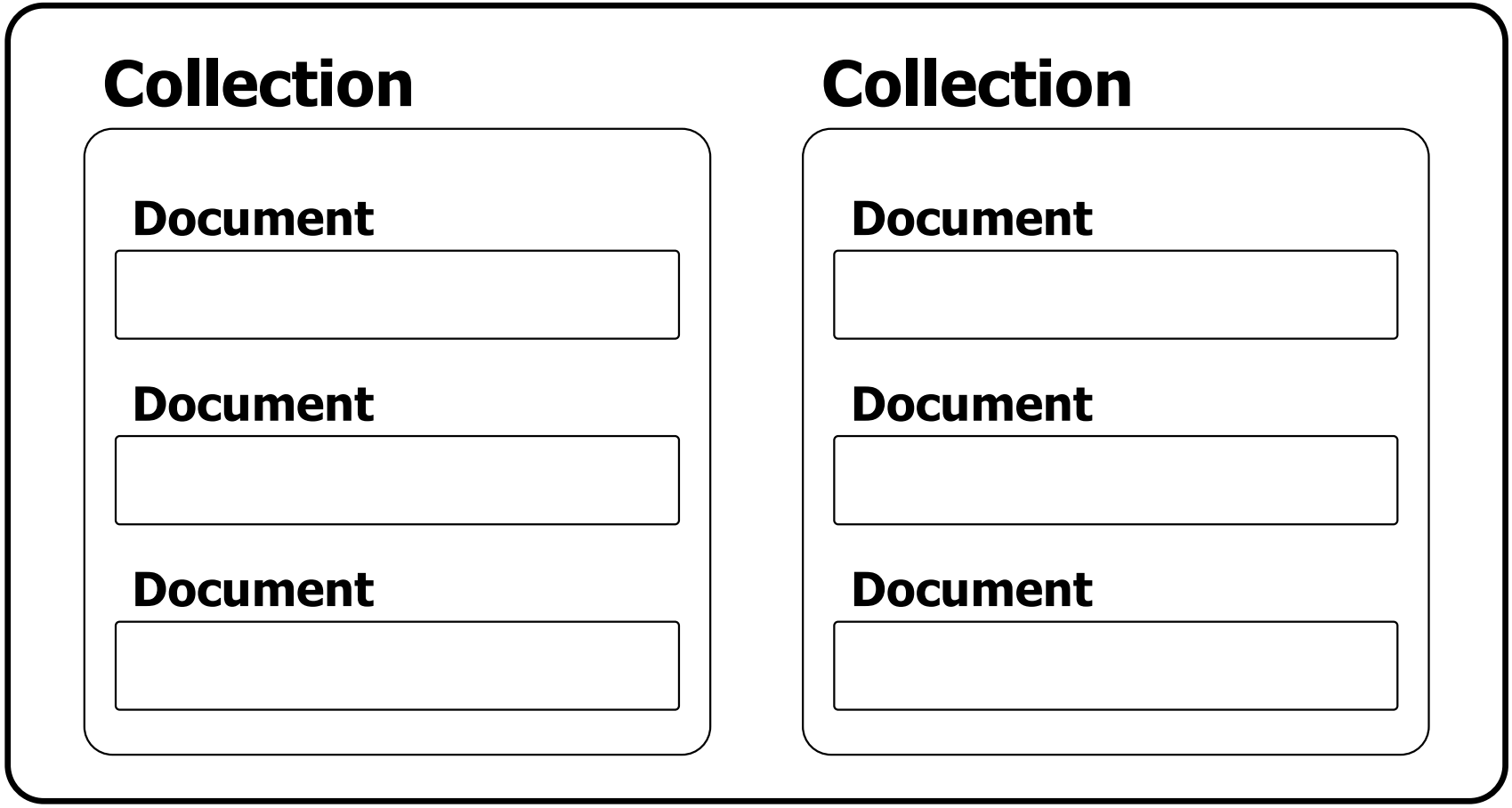
Array of actors

Collection

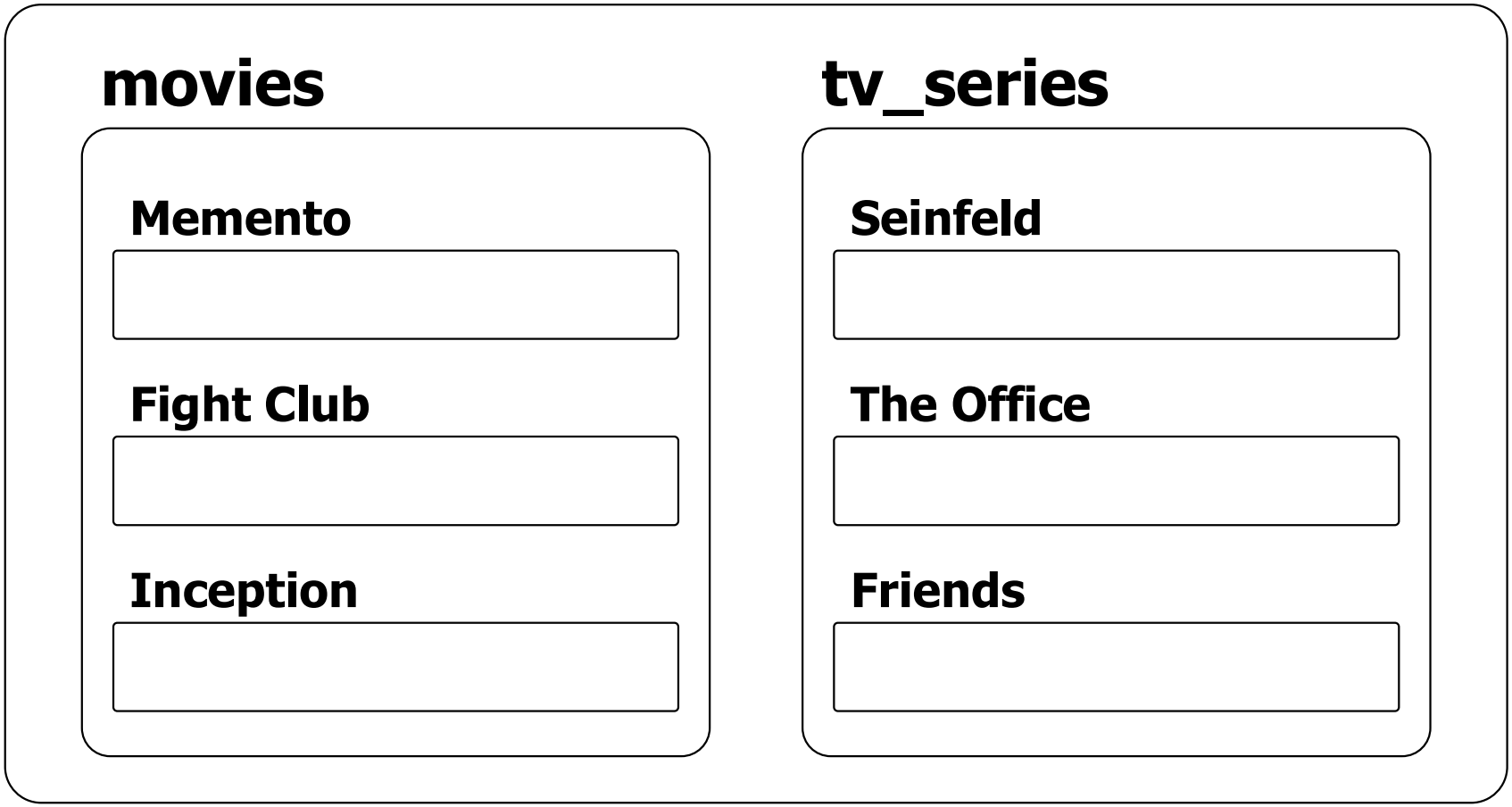
Bases de Datos de Documentos. Ejemplo

- *MongoDB*, de hu**mongo**us (gigantesco):
  - BD orientada a documentos multi-plataforma.
  - Evita la tradicional estructura de BDR (basada en tablas), utilizando documentos similares a JSON (llamada BSON), con esquemas dinámicos.
  - Con frecuencia para aplicaciones móviles, administración de contenido, análisis en tiempo real y aplicaciones relacionadas con el Internet de las cosas.
  - No está diseñado para datos transaccionales (sistemas contables por ejemplo).
  - Buena opción cuando no hay una definición de esquema clara.

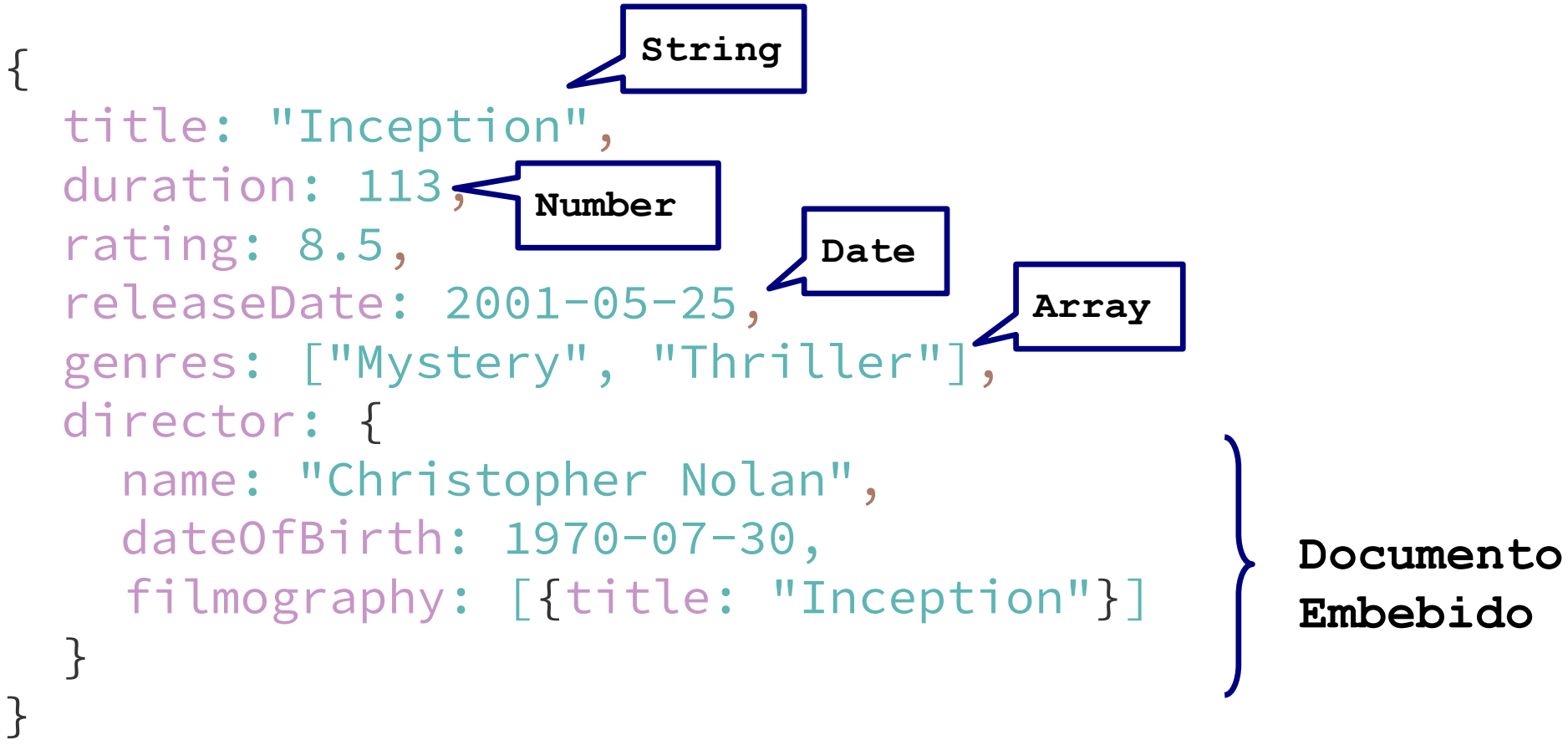
Database


Collection
Document
Document
Document
Collection
Document
Document
Document

my_imdb


movies
Memento
Fight Club
Inception
tv_series
Seinfeld
The Office
Friends

Document - Data Types



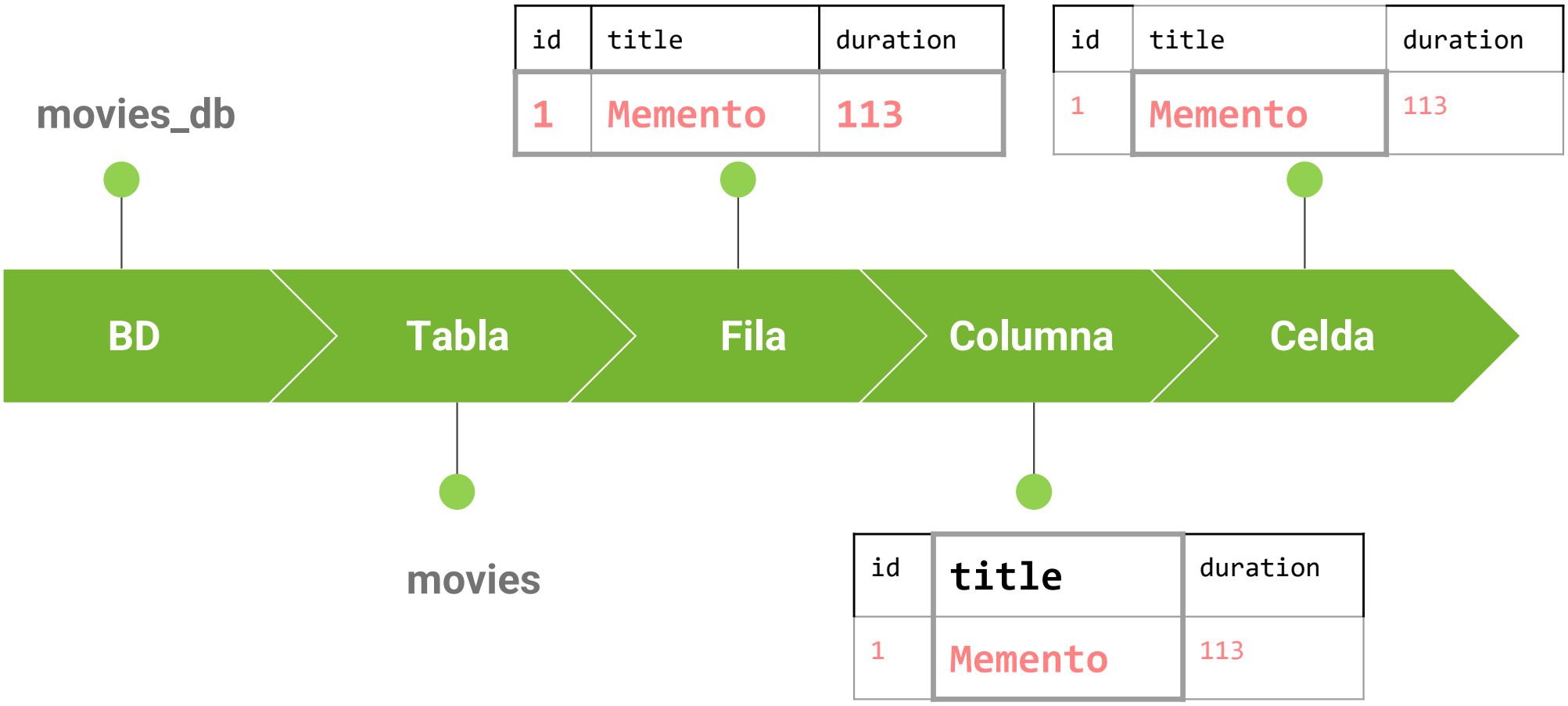

```
{
  title: "Inception",
  duration: 113,
  rating: 8.5,
  releaseDate: 2001-05-25,
  genres: ["Mystery", "Thriller"],
  director: {
    name: "Christopher Nolan",
    dateOfBirth: 1970-07-30,
    filmography: [{title: "Inception"}]
  }
}
```

Schema-less

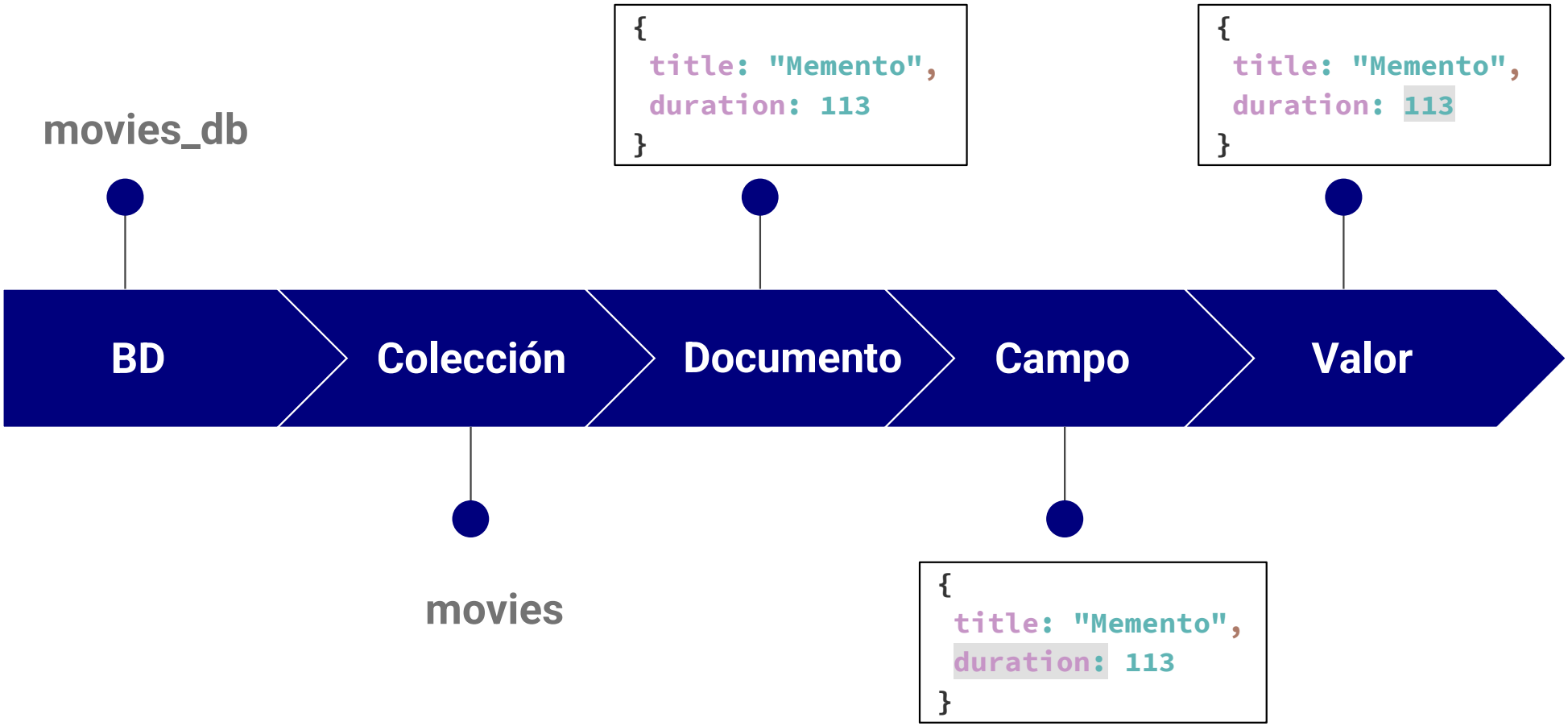
```
{
  title: "Memento",
  duration: 113,
  rating: 8.5,
  genres: [
    "Mystery",
    "Thriller"
  ]
}
```

```
{
  title: "Inception",
  duration: "148 min"
}
```

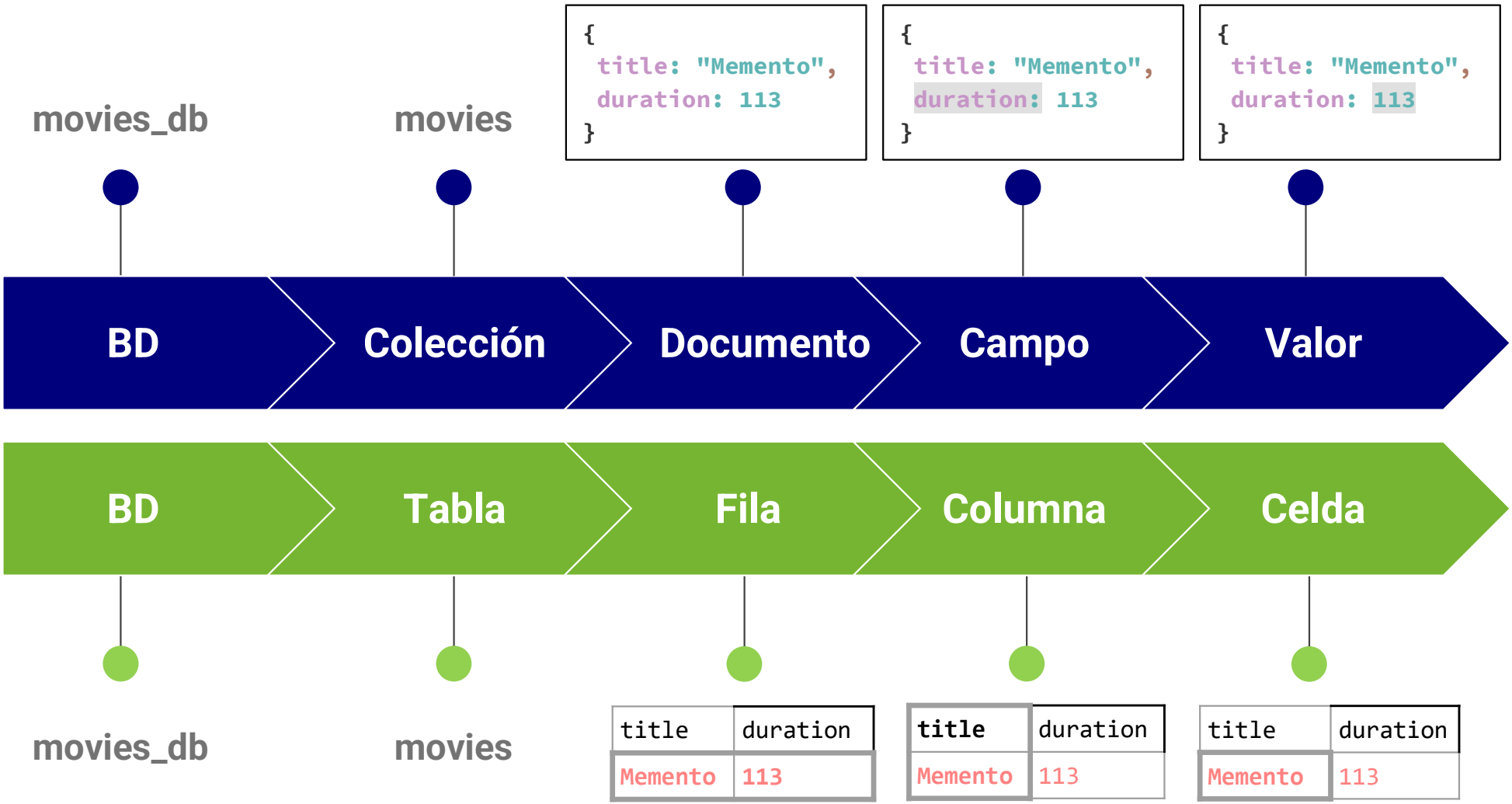
BDs Relacionales:



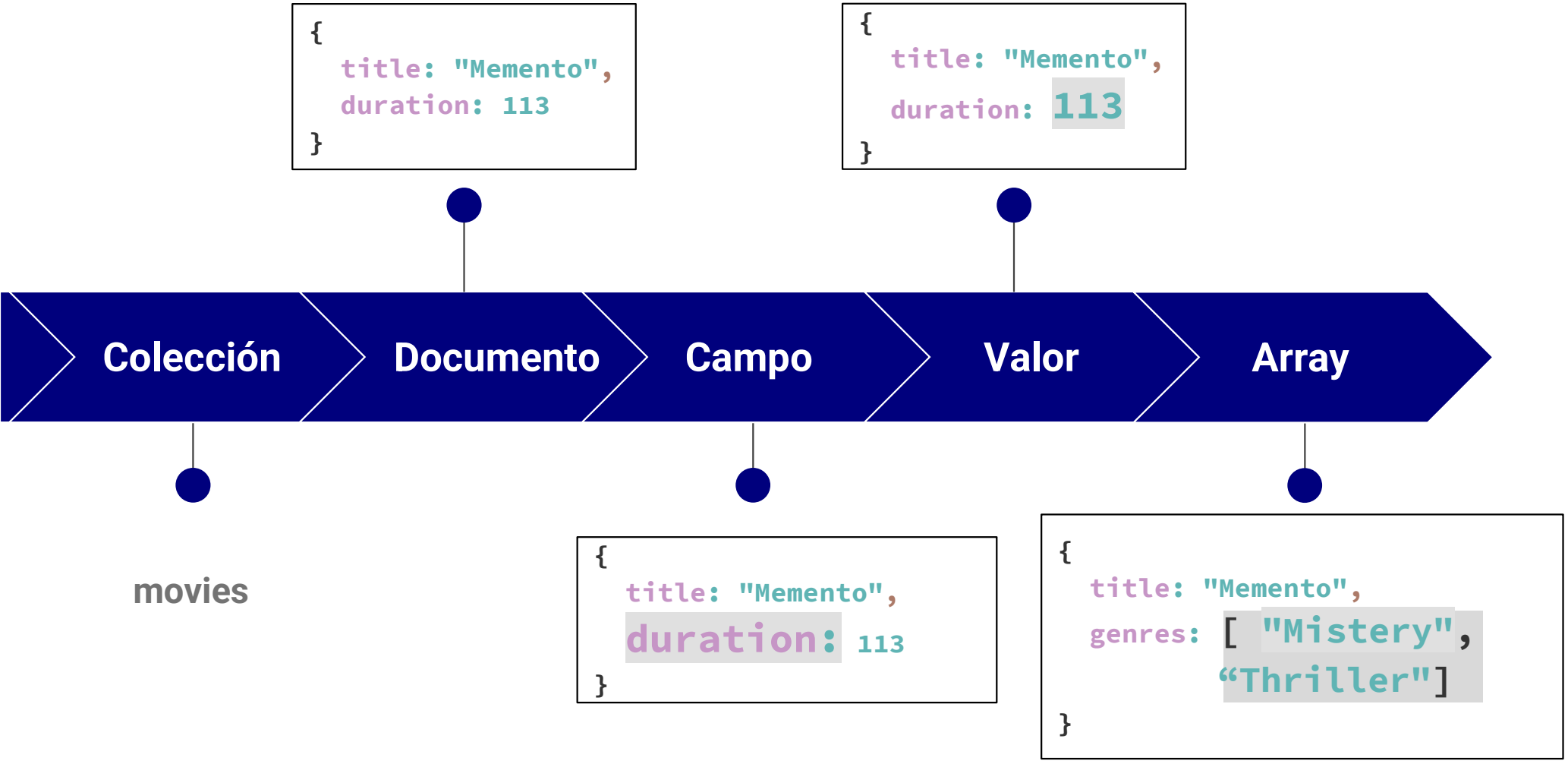
BDs Documentales:



BDs Rel - BDs Doc:



BDs Documentales:



Embebido vs Referenciado

```
{ title: "Inception",
  director: {
       name: "C. Nolan"
       dateOfBirth: 1970-07-30}
}

{ title: "Memento",
  director: {
       name: "C. Nolan"
       dateOfBirth: 1970-07-30}
}
```

```
{ title: "Inception",
  director_id: 1
}

{ title: "Memento",
  director_id: 1
}
```

Colección Directors:

```
{ id: 1,
name: "C. Nolan“
dateOfBirth: 1970-07-30
}
```

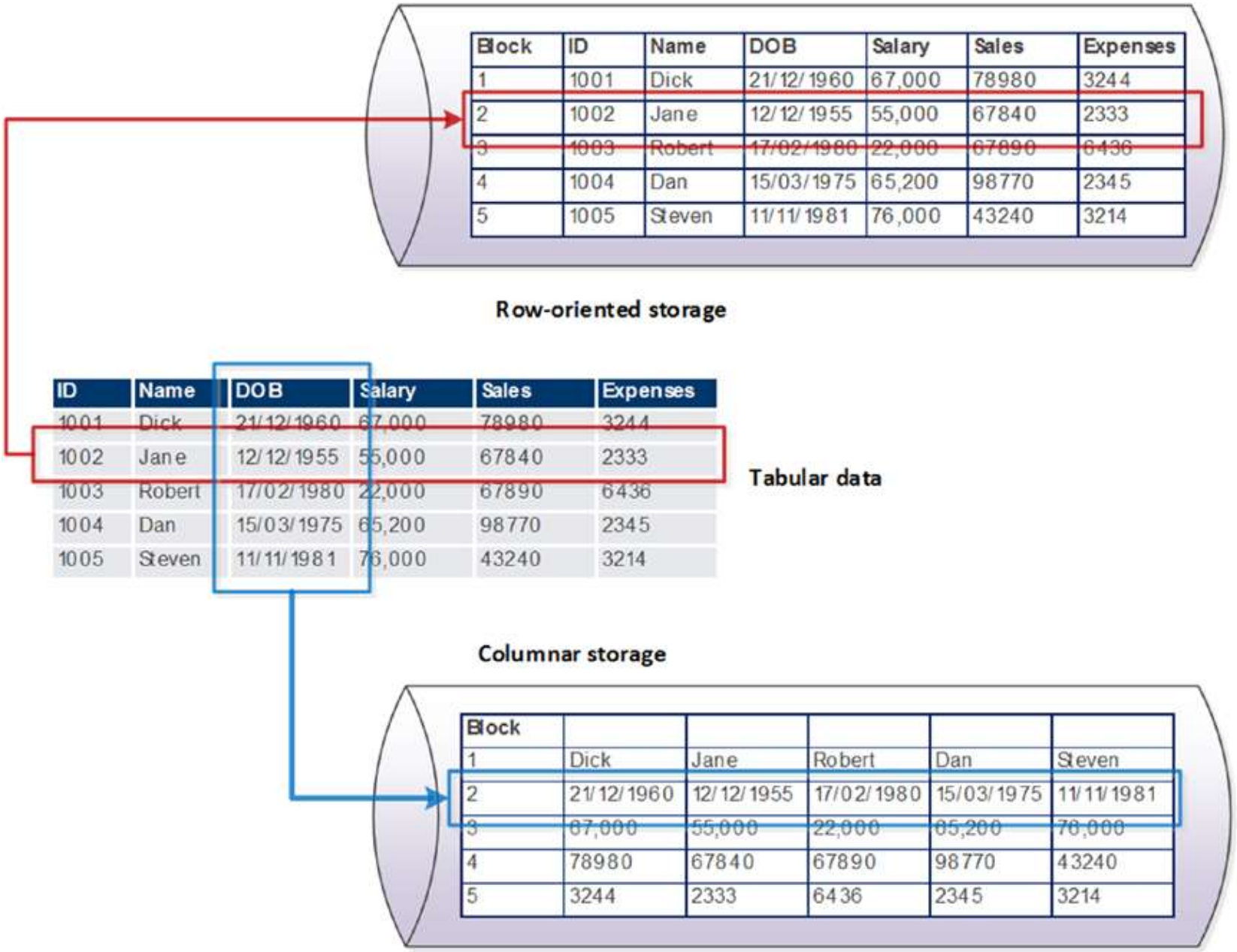
Bases de Datos Columnares

- En los Sistemas de DW rara vez se quiere procesar todas las columnas de una única fila. Generalmente, se desea procesar los valores de una sola columna en todas las filas.

- Las bases de datos columnares abordan esta necesidad.

- En esta arquitectura, los valores para cada columna (o atributo) son almacenados en disco de forma contigua.

Bases de Datos Columnares



Block	ID	Name	DOB	Salary	Sales	Expenses
1	1001	Dick	21/12/1960	67,000	78980	3244
2	1002	Jane	12/12/1955	55,000	67840	2333
3	1003	Robert	17/02/1980	22,000	67890	6436
4	1004	Dan	15/03/1975	65,200	98770	2345
5	1005	Steven	11/11/1981	76,000	43240	3214

Row-oriented storage

ID	Name	DOB	Salary	Sales	Expenses
1001	Dick	21/12/1960	67,000	78980	3244
1002	Jane	12/12/1955	55,000	67840	2333
1003	Robert	17/02/1980	22,000	67890	6436
1004	Dan	15/03/1975	65,200	98770	2345
1005	Steven	11/11/1981	76,000	43240	3214

Tabular data

Columnar storage

Block					
1	Dick	Jane	Robert	Dan	Steven
2	21/12/1960	12/12/1955	17/02/1980	15/03/1975	11/11/1981
3	67,000	55,000	22,000	65,200	76,000
4	78980	67840	67890	98770	43240
5	3244	2333	6436	2345	3214

Bases de Datos Columnares

- Idea clave: limitar la cantidad de datos a los que se accede por consulta, leyendo solo las columnas necesarias.

- El hecho de que las columnas se almacenen juntas en el mismo bloque de disco genera 2 ventajas importantes:
  - Las consultas que agregan los valores de una columna están optimizadas.
  - La compresión de los datos es mucho menos costosa computacionalmente.

Bases de Datos Columnares

- Desventaja:
  - La recuperación de una sola fila implica el ensamblaje de cada columna cargada para dicha fila en la tabla.
  - El *overhead* de lectura se puede reducir parcialmente:
    - mediante el almacenamiento en caché y
    - proyecciones de varias columnas (almacenando varias columnas juntas en el disco).

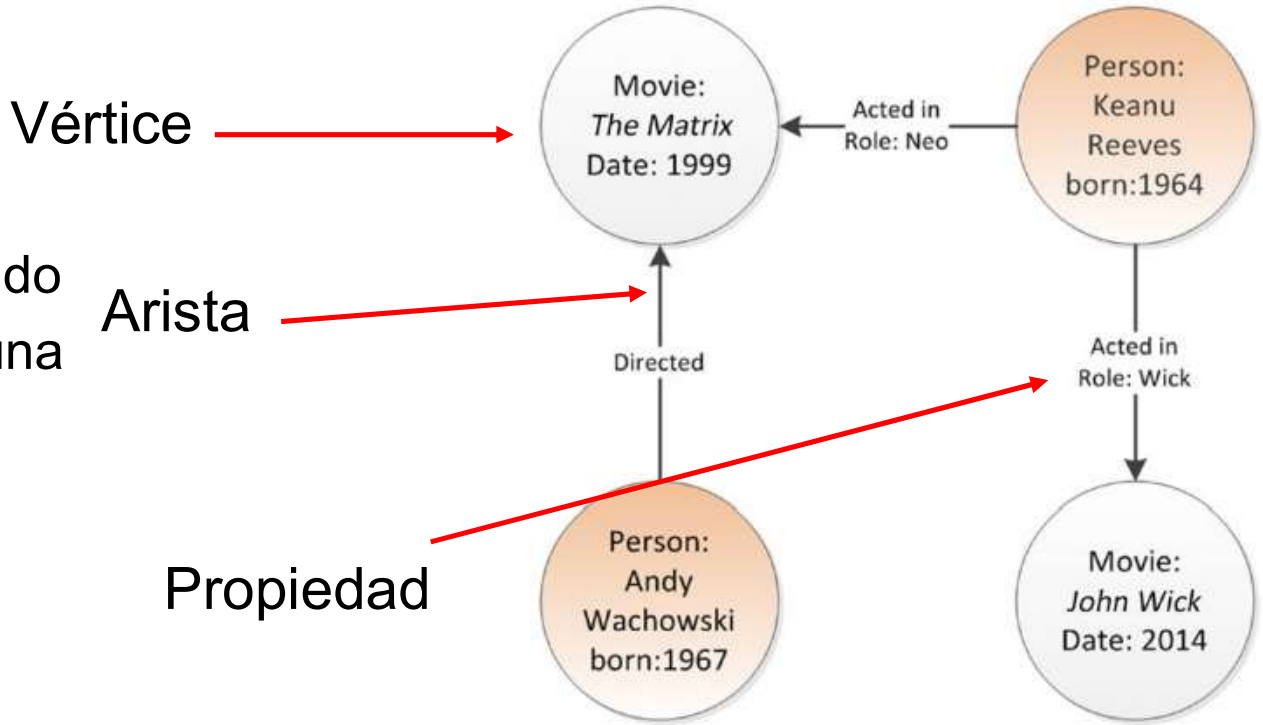
Bases de Datos Columnares. Ejemplo

- *Cassandra:*
  - No obliga a que todas las filas de una tabla tengan las mismas columnas.
  - A diferencia de los RDBMS tradicionales, las tablas se pueden crear, modificar y eliminar mientras se está ejecutando y procesando consultas.
  - Se accede a las columnas utilizando el lenguaje de consulta de *Cassandra* (CQL)
  - Sólo se puede consultar utilizando la clave primaria.
  - Fácil de configurar y mantener, independientemente de lo que crezca la BD.

Bases de Datos de grafos

- Almacenan estructuras de datos que tienen topología de grafo.
- Se basan en la teoría de grafos, la cual define los principales componentes de un grafo:
  - **Vértices** o nodos, que representan distintos objetos
  - **Aristas**, relaciones o arcos que conectan los objetos
  - Vértices y aristas pueden tener **propiedades**

Podríamos realizar un recorrido en el grafo para encontrar todos los actores que han aparecido como co-protagonistas en una película protagonizada por Keanu Reeves.



Bases de Datos de grafos

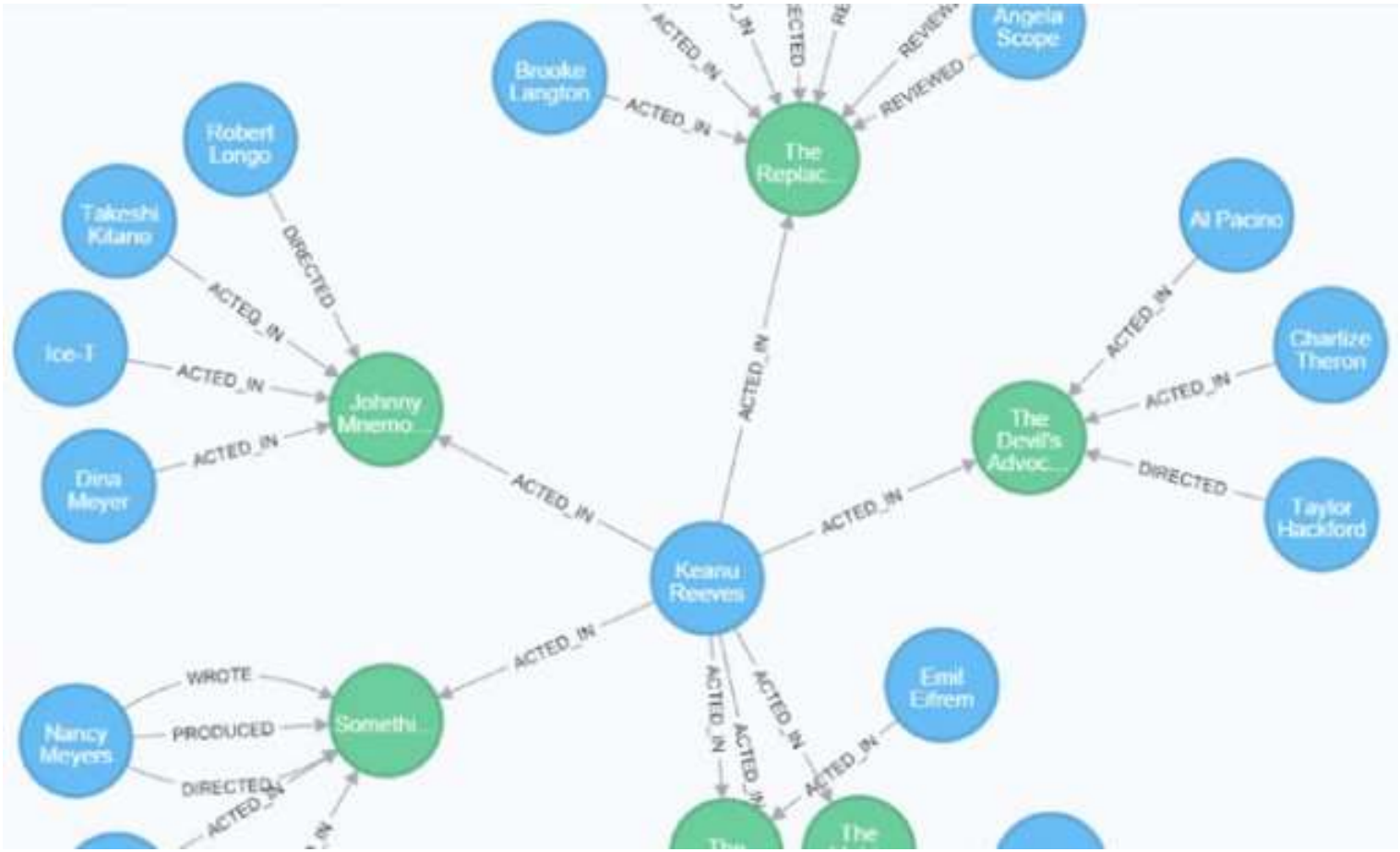
- Algunas bases de datos de grafos son internamente *key-value stores*, que agregan el concepto de las relaciones entre registros.
- El modelo de datos es flexible, lo que en algunos casos nos permite NO declarar tipos de vértices o de aristas.
  - El tipo de vértice o arista sería equivalente a las distintas tablas que se definen en un modelo relacional.
- Este tipo de BD facilita, en general, la exploración de datos, especialmente cuando las relaciones entre esos datos son tan significativas como los datos mismos.

Bases de Datos de grafos: Neo4j

- Neo4j es la base de datos de grafos más adoptada.
- Neo4j implementa un lenguaje de consulta de grafos declarativo: *Cypher*.
- Cypher permite que los grafos sean consultados utilizando una sintaxis simple comparable con SQL, pero particularmente optimiza los recorridos de los grafos.

Bases de Datos de grafos: Neo4j

- Cypher devuelve resultados que pueden ser grafos. Por ejemplo, la imagen es un grafo que muestra todos los vértices relacionados con "Keanu Reeves" a dos niveles.



Bibliografía

- Curso *Bases de Datos No Relacionales*: https://eva.fing.edu.uy/course/view.php?id=947
- Mike Stonebraker, Daniel J. Abadi, Adam Batkin, Xuedong Chen, Mitch Cherniack, Miguel Ferreira, Edmond Lau, Amerson Lin, Sam Madden, Elizabeth O'Neil, Pat O'Neil, Alex Rasin, Nga Tran, and Stan Zdonik. 2005. C-store: a column-oriented DBMS. In *Proceedings of the 31st international conference on Very large data bases* (VLDB '05). VLDB Endowment 553-564.
- Libro *Next Generation Databases*: https://link-springer-com.proxy.timbo.org.uy:88/book/10.1007%2F978-1-4842-1329-2
- MongoDB: https://www.mongodb.com/
- Cassandra: http://cassandra.apache.org/
- NOSQL: http://nosql-database.org/