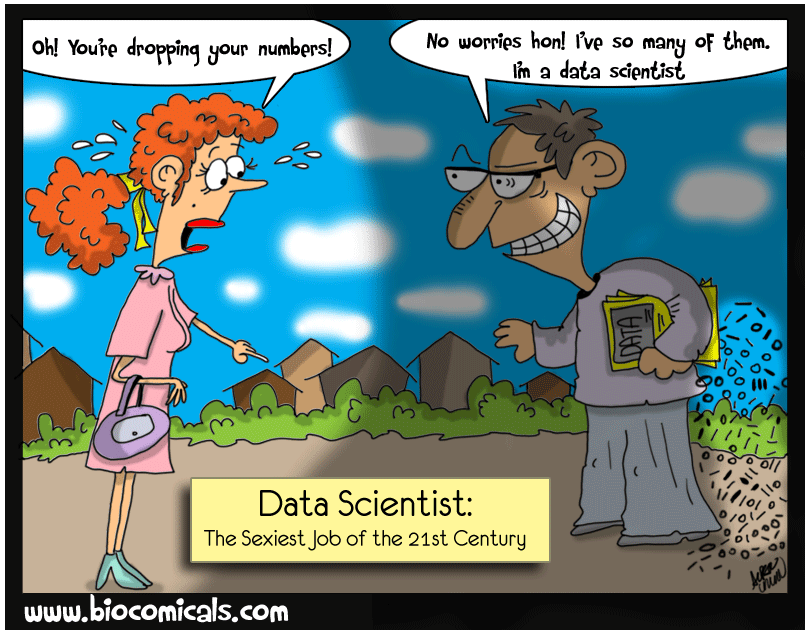
Otros modelos de datos

Bases de datos para Ingeniería, 2024

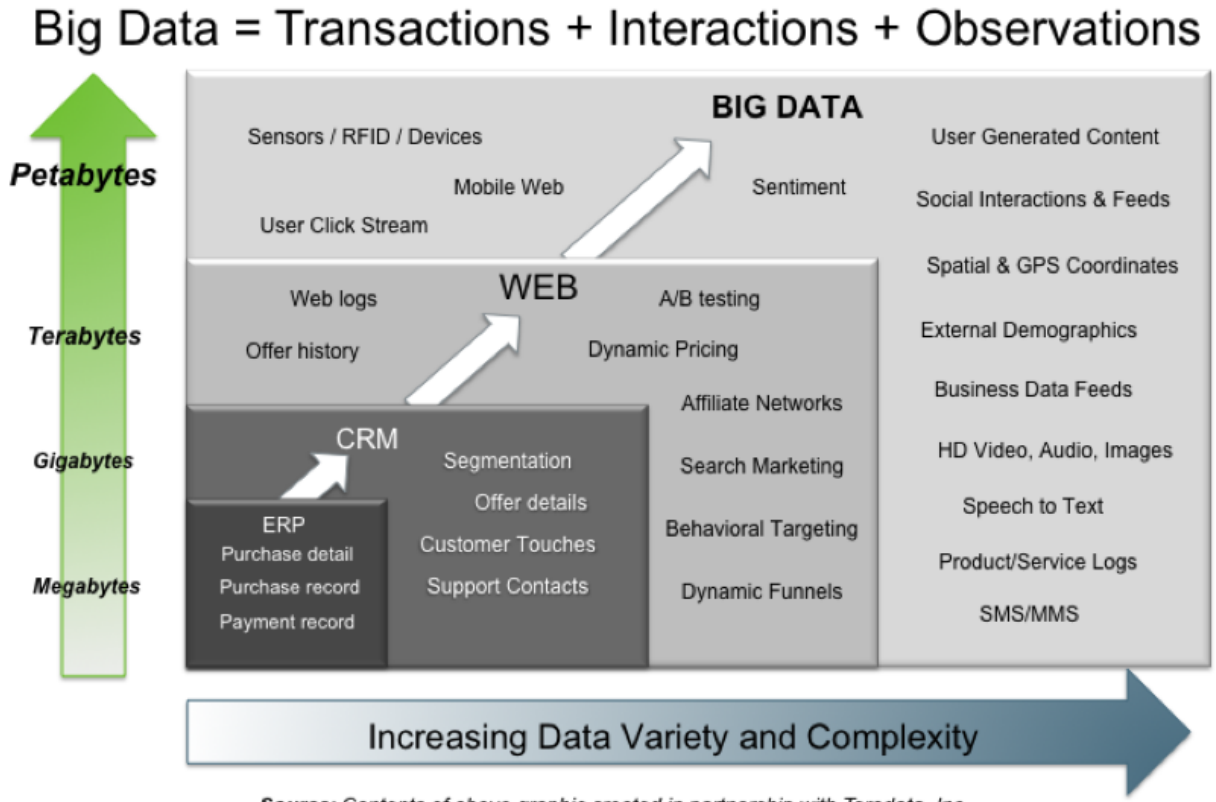
Lorena Etcheverry (lorenae@fing.edu.uy)
Instituto de Computación, FING, UdelaR

Recapitulemos

En los últimos 20 años el panorama es más complejo



La naturaleza de los datos y su volumen



***Source**: Contents of above graphic created in partnership with Teradata, Inc.*

Modelo de datos: definición

Los modelos de datos son **lenguajes** usados para especificar y manipular Bases de Datos

Un Modelo de Datos permite expresar:

- Estructuras: Elementos de los problemas.
- Restricciones: Reglas que deben cumplir los datos para que la base sea considerada válida.
- Operaciones: Insertar, borrar y consultar la BD.

Clasificación de los modelos de datos: nivel de abstracción

Conceptuales: Representan la realidad independientemente de cualquier implementación de la base de datos

Lógicos: Son implementados en un manejador de bases de datos particular.

Físicos: Corresponden a cómo está implementado el manejador de bases de datos (estructuras de datos)

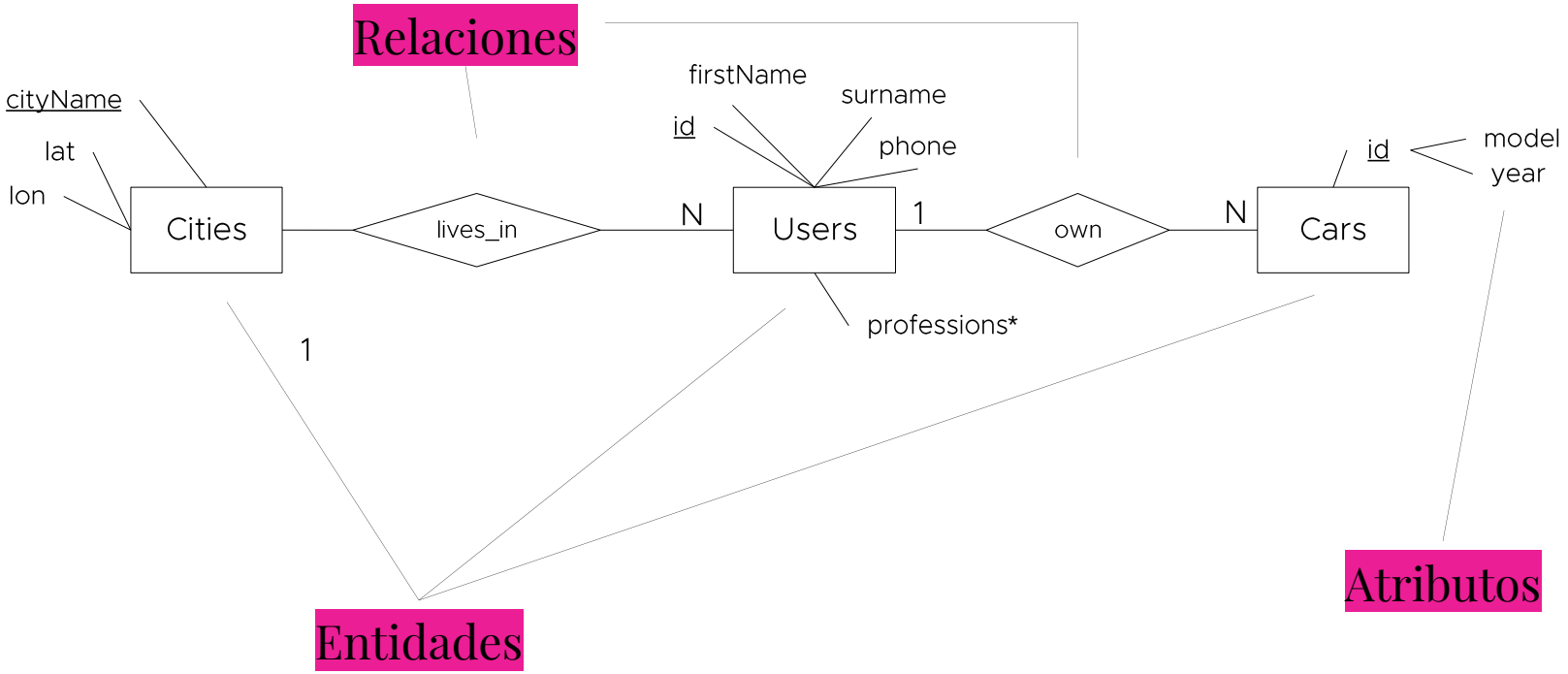
Modelos Conceptuales

En el contexto del curso vimos el modelo Entidad-Relación

> Queremos modelar una base de datos de propietarios de automóviles. De cada usuario se conoce su nombre y apellido, un número de teléfono y una o más profesiones. Cada usuario se identifica por un número de usuario, que es único. Se conoce además la ciudad en donde vive cada usuario, y para cada ciudad (identificada por su nombre) se conocen sus coordenadas (lat y long). Además interesa registrar los vehículos que posee cada usuario. Cada vehículo se identifica por el modelo y el año.

1- identifiquemos entidades (conceptos relevantes de la realidad)
2- identifiquemos atributos de las entidades
3- relaciones entre entidades y restricciones

Un posible diseño conceptual



Modelos lógicos

El **modelo Relacional** (Codd, 1970) es un ejemplo famoso y exitoso de modelo lógico.

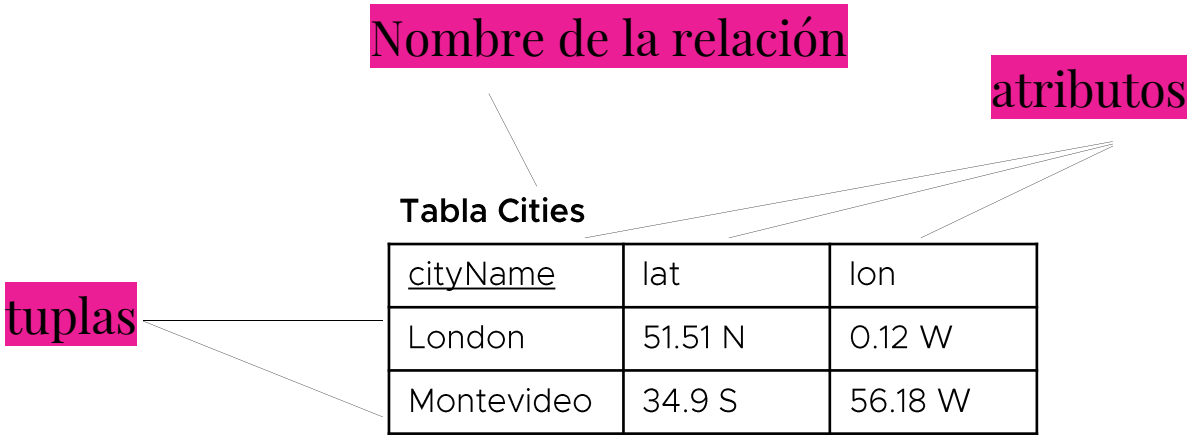
Los datos se representan como una colección de **relaciones** (tablas).

Cada relación tiene un nombre y un conjunto de atributos.

Las instancias de la relación (valores) son conjuntos de tuplas (sin repetidos, sin orden).

Además hay restricciones (ej: clave primaria)

Ejemplo Modelo Relacional



Ejemplo Modelo Relacional (cont)

Tabla Cities

cityName	lat	lon
London	51.51 N	0.12 W
Montevideo	34.9 S	56.18 W

Tabla Users

id	firstName	surname	phone	city
1	Paul	Miller	5111111	London
2	Laura	Rodriguez	333333	Montevideo

Tabla User_professions

userId	profession
1	banking
1	finance
1	trader

Tabla Cars

userId	model	year
1	Bentley	1973
1	Rolls Royce	1965

Diseño lógico de BDs relacionales

- Hay heurísticas que permiten obtener un diseño a partir del modelo conceptual, garantizando cierta calidad del diseño obtenido

–Pasaje de MER a ER

- Esta calidad es independiente de las consultas a realizar.

- Las Formas Normales garantizan propiedades para todas las instancias

Pero hay más modelos lógicos ...

- bases de datos documentales
- bases de datos de grafos

Bases de datos de documentos


```
{
  "data": [
    {
      "id": "X999_Y999",
      "from": {
        "name": "Tom Brady", "id": "X12"
      },
      "message": "Looking forward to 2010!",
      "actions": [
        {
          "name": "Comment",
          "link": "http://www.facebook.com/X999/posts/Y999"
        },
        {
          "name": "Like",
          "link": "http://www.facebook.com/X999/posts/Y999"
        }
      ],
      "type": "status",
      "created_time": "2010-08-02T21:27:44+0000",
      "updated_time": "2010-08-02T21:27:44+0000"
    }]
}
```

Una base de datos de documentos
es una base **no relacional**
(noSQL)
que almacena datos como
documentos estructurados,
típicamente
XML y **JSON**

XML: la vedette de los 2000s

Un lenguaje de marcado para representar datos.

XML 1.0 W3C recommendation en Febrero 1998.

¿para qué se pensó y para que se usa/usó?

Separar datos de presentación en la web 2.0

Agregar metadatos y esquema (dar estructura).

Intercambio de datos entre aplicaciones.

Interoperabilidad!

Estándares asociados a XML

XPath

Una sintaxis para recuperar partes del documento: filtros, comodines.

XQuery

Un lenguaje de consulta sobre XML. El SQL de XML!

XML Schema

Un documento XML especial que describe la estructura de otro documento XML

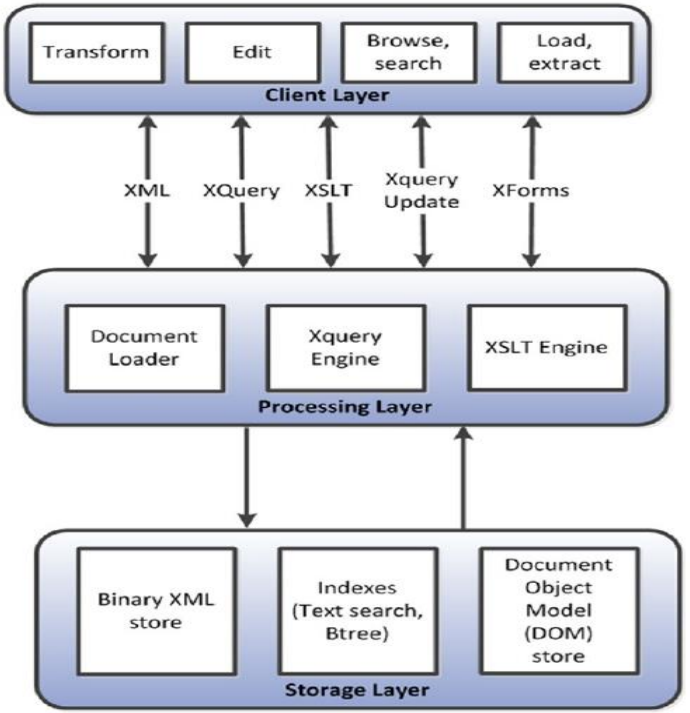
XSLT (eXtensible Stylesheet Language Transformations)

Un lenguaje de transformaciones. Permite generar otros formatos (ej: HTML)

Y como muchas aplicaciones generaban XML …

Surgen las BDs de documentos para gestionar colecciones de documentos XML.

Muchas soluciones comerciales.



pero XML tiene algunos inconvenientes...

```
<!DOCTYPE glossary PUBLIC "-//OASIS//DTD DocBook V3.1//EN">
 <glossary><title>example glossary</title>
  <GlossDiv><title>S</title>
   <GlossList>
    <GlossEntry ID="SGML" SortAs="SGML">
     <GlossTerm>Standard Generalized Markup
              Language</GlossTerm>
     <Acronym>SGML</Acronym>
     <Abbrev>ISO 8879:1986</Abbrev>
     <GlossDef>
      <para>A meta-markup language, used to create markup
          languages such as DocBook.</para>
      <GlossSeeAlso OtherTerm="GML">
      <GlossSeeAlso OtherTerm="XML">
     </GlossDef>
     <GlossSee OtherTerm="markup">
    </GlossEntry>
   </GlossList>
  </GlossDiv>
 </glossary>
```

"Some languages can be read by humans, but not by machines,

while others can be read by machines but not by humans.

XML solves this problem by being readable to neither.“

[1] https://www.reddit.com/r/ProgrammerHumor/comments/1mbgih/xml/

Mientras tanto, en el mundo de los desarrolladores de aplicaciones web ...

Javascript se impone como lenguaje de programación tanto del lado del cliente como del servidor (ej: Node.js).

AJAX: mensajería asíncrona entre cliente y servidor (originalmente pensado sobre XML).

Aparece Javascript Object Notation (JSON) como mecanismo para serializar objetos.

JSON: The Fat-Free alternative to XML [1]

[1]http://www.json.org/xml.html

Estructura de un doc JSON

objeto

clave

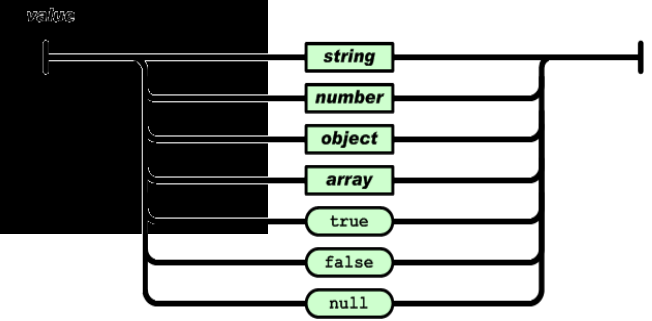
valor

arreglo (lista de valores)

```
{ "glossary": {
    "title": "example glossary",
    "GlossDiv": {
      "title": "S",
      "GlossList": {
        "GlossEntry": {
          "ID": "SGML",
          "SortAs": "SGML",
          "GlossTerm": "Standard Generalized Markup Language",
          "Acronym": "SGML",
          "Abbrev": "ISO 8879:1986",
          "GlossDef": {
            "para": "A meta-markup language, used to
                create markup languages such as DocBook.",
            "GlossSeeAlso": ["GML", "XML"]
          },
          "GlossSee": "markup"
        }
      }
    }
  }
}
```

Las *claves* son *strings*.

Los *valores* pueden ser cualquier elemento de la notación.



XML vs JSON (1)


xml
Término de búsqueda
json
Término de búsqueda
+ Añadir comparación
Todo el mundo
2004 - hoy
Todas las categorías
Búsqueda web
Interés a lo largo del tiempo
100
75
50
25
Media
1 ene 2004
1 mar 2011
mar 2022
xml 10
json 18
Los valores de este punto tienen datos parciales (incompletos)

XML vs JSON (2)

```
<!DOCTYPE glossary PUBLIC "-//OASIS//DTD DocBook V3.1//EN">
 <glossary><title>example glossary</title>
  <GlossDiv><title>S</title>
   <GlossList>
    <GlossEntry ID="SGML" SortAs="SGML">
     <GlossTerm>Standard Generalized Markup
              Language</GlossTerm>
     <Acronym>SGML</Acronym>
     <Abbrev>ISO 8879:1986</Abbrev>
     <GlossDef>
      <para>A meta-markup language, used to create markup
           languages such as DocBook.</para>
      <GlossSeeAlso OtherTerm="GML">
      <GlossSeeAlso OtherTerm="XML">
     </GlossDef>
     <GlossSee OtherTerm="markup">
    </GlossEntry>
   </GlossList>
  </GlossDiv>
```

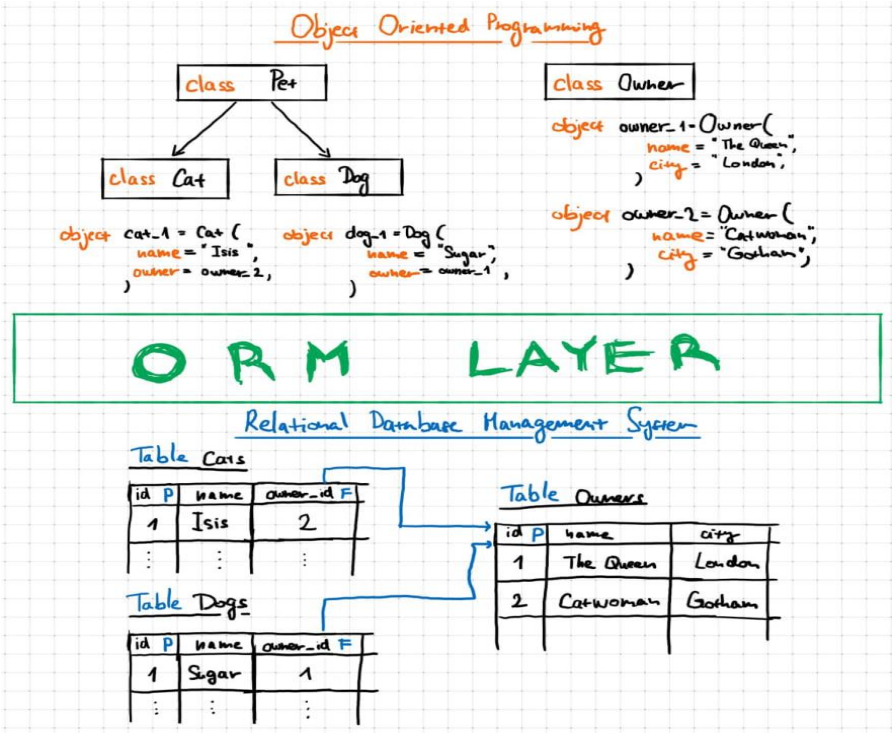
```
{ "glossary": {
    "title": "example glossary",
    "GlossDiv": {
      "title": "S",
      "GlossList": {
        "GlossEntry": {
          "ID": "SGML",
          "SortAs": "SGML",
          "GlossTerm": "Standard Generalized Markup                    Language",
          "Acronym": "SGML",
          "Abbrev": "ISO 8879:1986",
          "GlossDef": {
            "para": "A meta-markup language, used to
                    create markup languages such as
                    DocBook.",
            "GlossSeeAlso": ["GML", "XML"]
          },
          "GlossSee": "markup"
        }
      }
    }
  }
}
```

Algunas ventajas de JSON

- JSON es **menos verboso**: más liviano para el intercambio de datos y más rápido su parseo.
- JSON representa **directamente** el objeto: más sencillo para el desarrollador
- No tengo un esquema fijo (salvo q use JSON Schema), pero tengo **algo de estructura**.
- Si desarrollo en Javascript, intercambio JSON y almaceno JSON tengo un ***full stack* Javascript**.

- Si desarrollo OO y almaceno JSON **no tengo *impedance mismatch*** entre el modelo OO y el relacional.

Object- relational impedance mismatch



Describe la dificultad que surge al tratar de integrar sistemas de bases de datos relacionales con lenguajes de programación orientados a objetos.

Algunos “problemas”

1. problemas de granularidad

2. problema de la herencia

3. problema de la identidad

4. problema de la asociación y navegación de datos

Ireland, C., & Bowers, D. (2015, May). Exposing the myth: object-relational impedance mismatch is a wicked problem. In DBKDA 2015, The Seventh International Conference on Advances in Databases, Knowledge, and Data Applications (pp. 21-26). IARIA XPS Press.

Diseño de bases de datos de documentos

BD relacional

BD documental

Tabla	Colección
Fila (tupla)	Documento
Columna (atributo)	Campo (*field*)
SQL Join	Documentos embebidos y referencias
SQL Group-by (agregación)	*Aggregation pipeline*

- Los documentos dentro de una colección pueden tener campos diferentes (diferente esquema).
- Se *puede* exigir la validación de cierto esquema definido con JSON Schema

–Ejemplos de esquemas

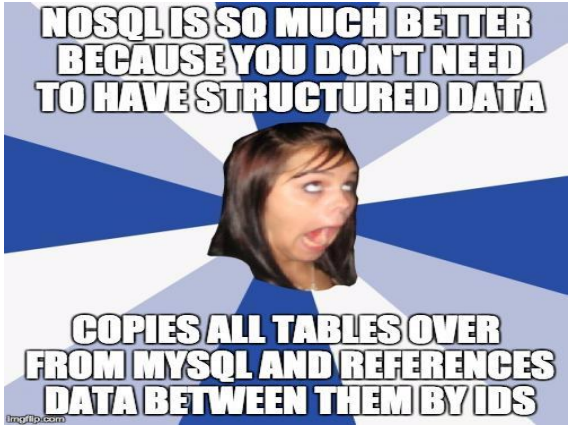
La ausencia de un esquema fijo (*schema-less*) no quiere decir que no exista una etapa de diseño y modelado

Diseño de BDs documentales

- No hay heurísticas que permitan pasar desde conceptual a documental como en el caso relacional que garanticen la calidad de la solución

- El diseño depende del tipo de operaciones que voy a hacer además de la naturaleza de los datos

- Diseñar “a la relacional” no suele ser lo mejor



Diseño de BDs documentales

¿Cuáles son las variables que tenemos?

–Decidir cómo se van a representar los elementos de la realidad.

–Decidir cómo representar las relaciones entre elementos.

Estrategias para representar relaciones


```
{
  "imdbID":"tt0944947",
  "Type":"series",
  "Title":"Game of Thrones",
  "Genre":"Adventure, Drama, Fantasy",
  "Actors":"Peter Dinklage, Lena Headey, Emilia Clarke, Kit Harington",
  "imdbRating":"9.5",
  "imdbVotes":"1,031,056"
}
{
  "imdbID":"tt1877832",
  "Type":"movie"
  "Title":"X-Men: Days of Future Past",
  "Year":"2014",
  "Genre":"Action, Adventure, Fantasy",
  "Actors":"Hugh Jackman, Michael Fassbender, Jennifer Lawrence, Peter Dinklage
  "imdbRating":"8.0",
  "imdbVotes":"514,203"
```


Estrategia 1:

documentos embebidos

```
{
 "imdbID":"tt0944947",
 "Type":"series",
 "Title":"Game of Thrones",
 "Genre":"Adventure, Drama, Fantasy",
 "Actors":
    [
      { “imdbID”: “nm0227759” , “name”: "Peter Dinklage”},
      { “imdbID”: “nm0372176” , “name”: "Lena Headey”},
      { “imdbID”: “nm3229685” , “name”: "Kit Harington"}
                          ]
 "imdbRating":"9.5",
 "imdbVotes":"1,031,056"
}
{
 "imdbID":"tt1877832",
 "Type":"movie"
 "Title":"X-Men: Days of Future Past",
 "Year":"2014",
 "Genre":"Action, Adventure, Fantasy",
 "Actors":
    [
      { “imdbID”: “nm0413168” , “name”: "Hugh Jackman”},
      { “imdbID”: “nm1055413”, “name”: "Michael Fassbender”},
      { “imdbID”: “nm2225369”, “name”: "Jennifer Lawrence”},
      { “imdbID”: “nm0227759” , “name”: "Peter Dinklage”}
                          ]
 "imdbRating":"8.0",
 "imdbVotes":"514,203"
}
```

Documentos embebidos

VENTAJAS

- Recupero toda la información relevante en una sola consulta.

– Evita implementar joins en el código de la aplicación o utilizar $lookup.

- Actualizo la información relacionada como una única operación atómica.

– Por defecto, todas las operaciones CRUD sobre un mismo documento son compatibles con ACID.

LIMITACIONES

- Duplicación de datos

- Los documentos grandes suponen más sobrecarga si la mayoría de los campos no son relevantes.

– Se puede aumentar el rendimiento de las consultas limitando el tamaño de los docs

- MongoDB tiene un límite de tamaño de documento de 16 MB.

– Si estoy colocando demasiados datos dentro de un único documento, podría alcanzar este límite.

Estrategia 2:
documentos referenciados

```
{
  "imdbID":"tt0944947",
  "Type":"series",
  "Title":"Game of Thrones",
  "Genre":"Adventure, Drama, Fantasy",
  "Actors":
      ["nm0227759", "nm0372176", "nm3229685" ]
  "imdbRating":"9.5",
  "imdbVotes":"1,031,056"
}
{
  "imdbID":"tt1877832",
  "Type":"movie"
  "Title":"X-Men: Days of Future Past",
  "Year":"2014",
  "Genre":"Action, Adventure, Fantasy",
  "Actors":
      [ "nm0413168" ,"nm1055413","nm2225369","nm0227759"]
  "imdbRating":"8.0",
  "imdbVotes":"514,203"
}
...
{ "id":"nm0227759",
  "name":"Peter Dinklage",
```

Documentos referenciados

VENTAJAS

- Evito duplicación de datos
- Menor tamaño en los documentos
- La información a la que se accede con poca frecuencia no se retorna en cada consulta.

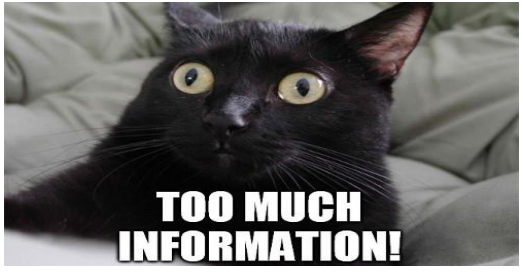
LIMITACIONES

- Para recuperar toda la información relevante debo utilizar $lookup.
- La actualización puede impactar a más de un documento (transacción).
- Aunque el operador de transacción está disponible desde la versión 4.0, es un anti-patrón depender demasiado de su uso.

Aspectos a tener en cuenta en el diseño

- Atomicidad de las operaciones (ACID a nivel de documento)
- Restricción de tamaño máximo de cada documento.
- Restricciones de tamaño y cantidad de documentos en las colecciones
- Operaciones a realizar y su frecuencia
- Ciclo de vida del dato (permanente, volátil?)
- Necesidades de particionamiento (sharding)

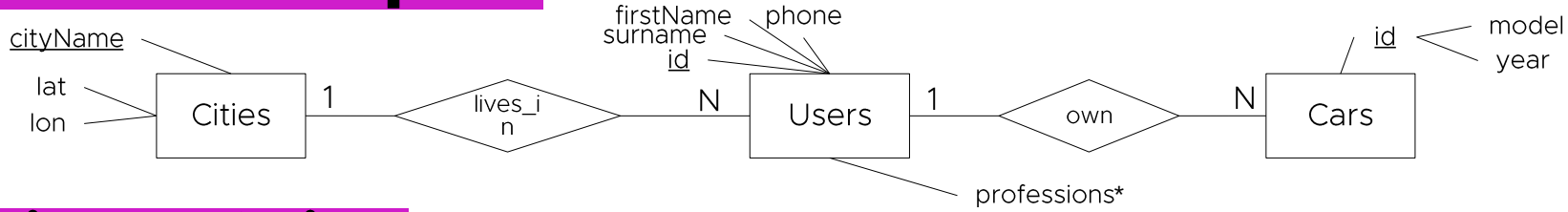
Estos aspectos condicionan los diseños posibles



¿Cómo combinamos todos estos ingredientes?

- Aplicando recomendaciones básicas de diseño
- Considerando patrones de diseño
- Usando metodologías emergentes que buscan optimizar costos

Diseño conceptual



Diseño lógico relacional

	loc_x	loc_y
London	51.51 N	0.12 W
Montevideo	34.9 S	56.18 W

Tabla User_professions

	profession
1	banking
1	finance
1	trader

Tabla Users

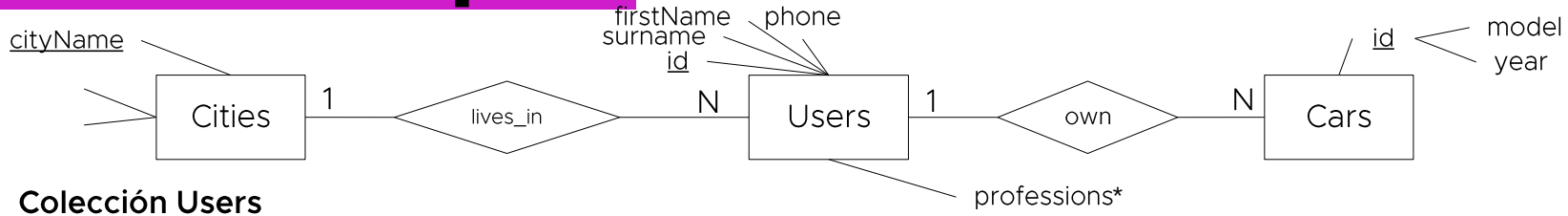
	firstName	surname	phone	city
1	Paul	Miller	5111111	London

Tabla Cars

	model	year
1	Bentley	1973
1	Rolls Royce	1965

Ahora construyamos un diseño lógico documental, asumiendo que las consultas están centradas en usuarios

Diseño conceptual



Colección Users

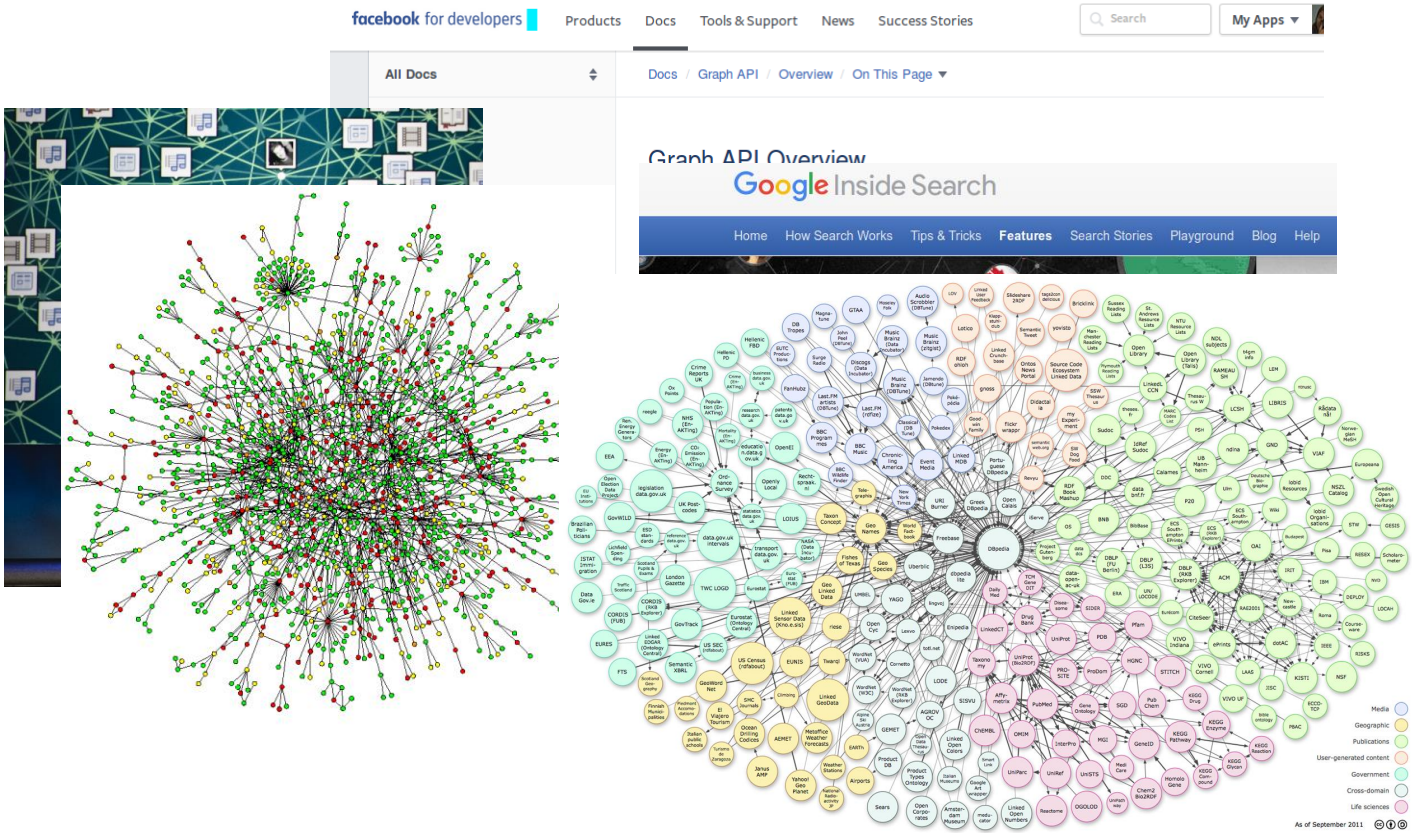
```
{
   "first_name": "Paul",
   "surname": "Miller",
   "cell": "5111111",
   "city": "London",
   "location": [51.51, 0.12],
   "profession": ["banking", "finance", "trader"],
   "cars": [
      {
         "model": "Bentley",
         "year": 1973
      },
      {
         "model": "Rolls Royce",
         "year": 1965
      }
   ]
}
```

- Relaciones 1:1 – Preferir parejas clave valor en el documento

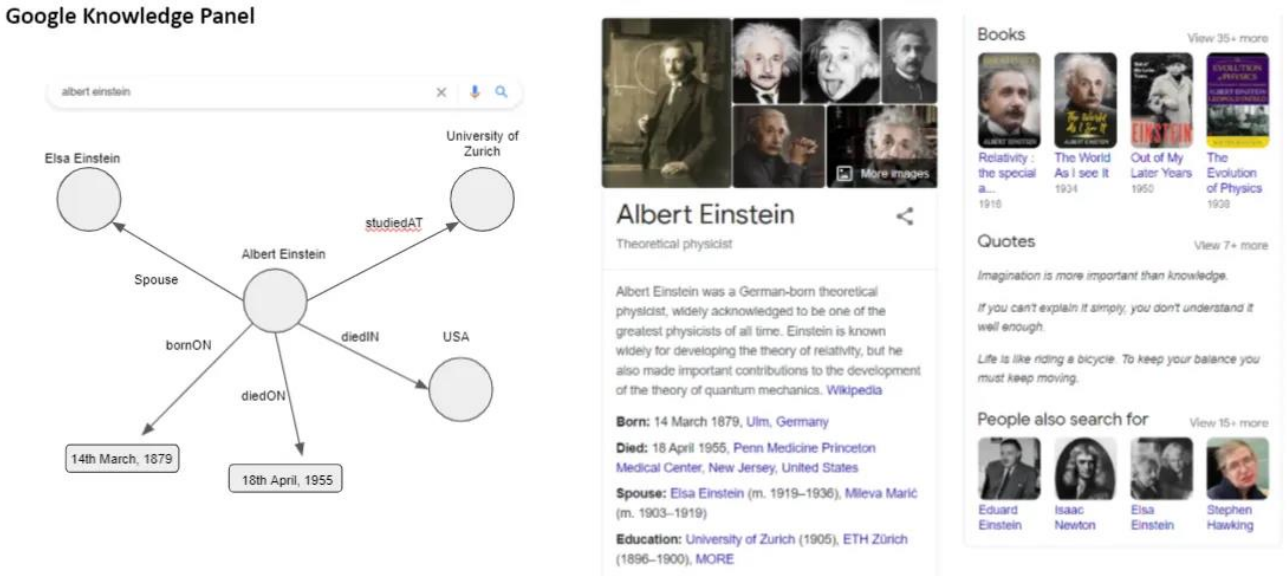
- Relaciones 1:N
- 1 a pocos – Preferir documentos embebidos

Bases de datos de grafos

Hay casos en que interesa más modelar las **relaciones** entre cosas que las cosas!



Knowledge Graphs



Fuente: https://towardsdatascience.com/a-guide-to-the-knowledge-graphs-bfb5c40272f1

¿en qué escenarios son adecuadas las BD de grafos?

- Las entidades están muy conectadas a través de relaciones descriptivas.
- Existen relaciones cíclicas o entidades autorreferenciadas.
    - Esto suele ser un reto cuando se utilizan bases de datos relacionales o documentales.
- Las relaciones entre entidades evolucionan dinámicamente.
    - especialmente aplicable a datos jerárquicos o estructurados en árbol con muchos niveles.
- Existen relaciones de muchos a muchos entre las entidades.
- Existen requisitos de escritura y lectura tanto en las entidades como en las relaciones.

Modelos de grafos en bases de datos de grafos

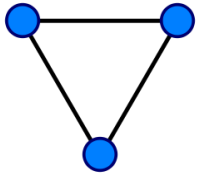
¿Qué es un grafo?

Un grafo G consiste en
un conjunto de nodos o vértices V,
y un conjunto de aristas E.
Las aristas conectan nodos entre si.

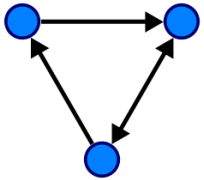
Las bases de datos de grafos
implementan diferentes variantes[1].

[1] *Survey of Graph Database Models*, Angles and Gutierrez,
ACM Computing Surveys, 2008

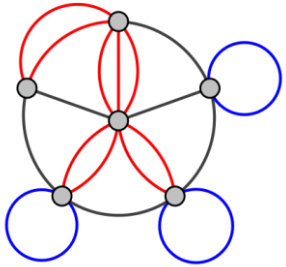
Grafos según su estructura



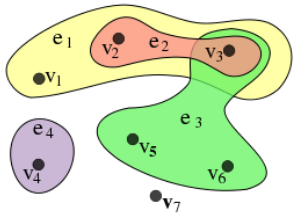
Grafo no dirigido



Grafo dirigido

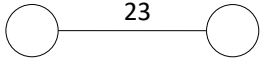


Multigrafos y pseudografos



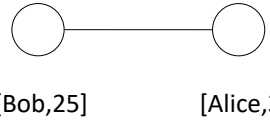
Hipergrafos

Grafos según los datos asociados



Etiquetas en las aristas

Etiquetas en los nodos



Atributos en los nodos

¿qué tipos de grafos se utilizan en las bases de datos de grafos?

Existen muchos modelos

(ver *Survey of Graph Database Models*, Angles and Gutierrez, ACM Computing Surveys, 2008)

Los sistemas más populares
actualmente implementan el modelo
***property graph* o RDF**

Property Graph Model (PGM)

Pseudografos dirigidos.

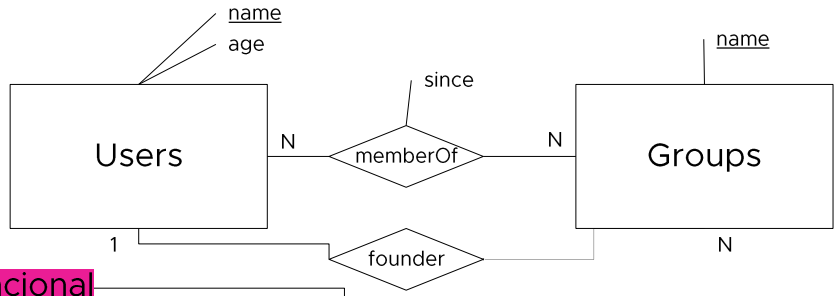
Parejas (clave,valor) llamadas *propiedades* asociadas a nodos y aristas.

Además puedo etiquetar nodos y aristas.

Cada nodo o arista puede tener más de una etiqueta

Neo4j y Titan son ejemplos de sistemas de bases de datos que soportan PGM.

CONCEPTUAL



LÓGICO

Relacional

Users

name	**age**
Bob	25
Alice	null

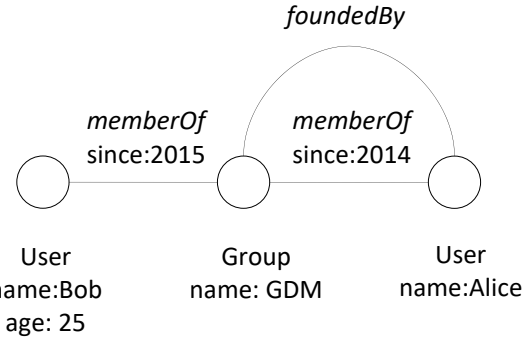
Groups

name	**founder**
GDM	Alice

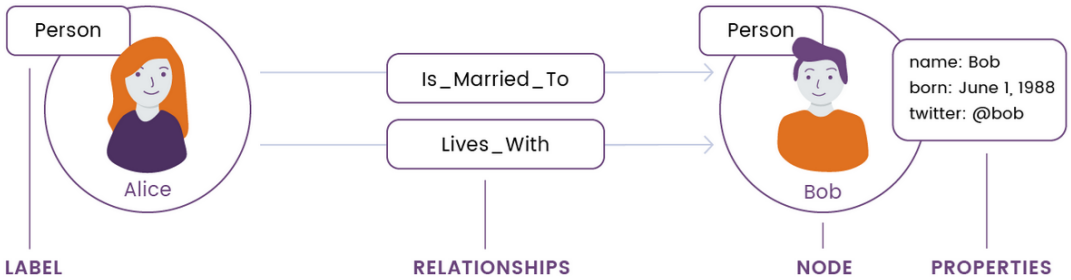
GroupMembers

userName	**groupName**	**since**
Bob	GDM	2015
Alice	GDM	2014

Property Graphs



Otro ejemplo



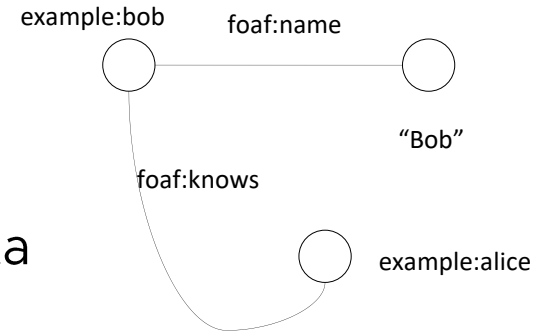
Resource Description Framework (RDF)

Pseudografos dirigidos.

Etiquetas en nodos y aristas.

Las etiquetas pueden ser:
IRIs, literales, o nodos blancos.

Cada nodo o arista tiene una sola etiqueta



Virtuoso y Stardog son ejemplos de sistemas de bases de datos que soportan RDF.

Restricciones y esquemas

- En ambos casos es posible definir nociones de esquema más o menos fuertes.
- En la familia de lenguajes y modelos vinculados a RDF puedo hacer cosas muy expresivas y potentes.
- Sobre el PGM puedo definir algunas restricciones pero dependen del vendedor
  - Ejemplo Neo4j constraints

Grafos de conocimiento y la web semántica

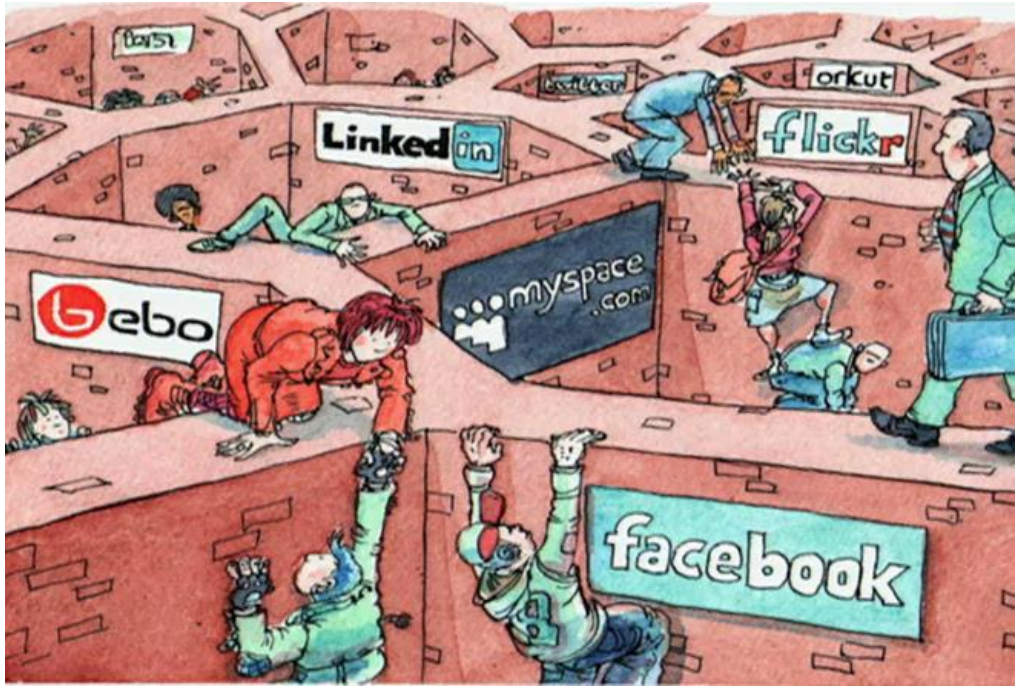
La web clásica es una red de **documentos**,

interpretable por **humanos**,

donde las relaciones entre documentos no tienen un

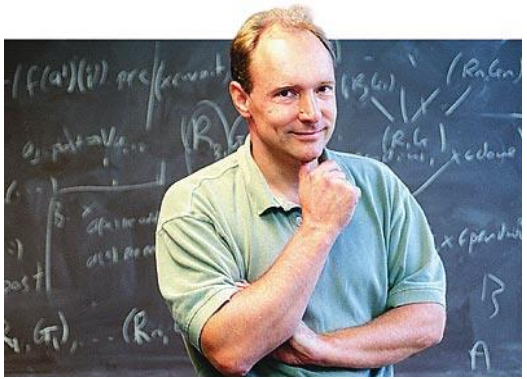
significado.

Además, la mayoría de los datos en la web están a i s l a d o s



La web de datos es una red de **afirmaciones**, interpretable por humanos y **máquinas**, donde las afirmaciones se relacionan y tienen un **significado**.

W3C Semantic Web



RDF: un modelo de datos basado en triplas que permite representar relaciones.

SPARQL: el lenguaje de consultas sobre RDF.

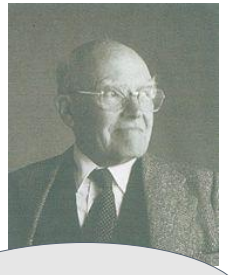
RDF-S y OWL, para representar metadatos y darles significado (ontologías).

http://www.w3.org/2001/sw/wiki/RDF

http://www.w3.org/2001/sw/wiki/SPARQL

afirmación o tripla

<sujeto, predicado, objeto>



Ing. Eladio Dieste

esAutor

MVD Shopping Center

afirmación o tripla

<sujeto, predicado, objeto>

En RDF todo es una tripla.

Los recursos y las propiedades se identifican por URIs

Linked Data

es un conjunto de
buenas prácticas para
publicar y *relacionar*
datos en la web,
usando
tecnologías de la
web semántica

Principios de Linked Data

1) Usar URIs para nombrar cosas

2) Usar URIs HTTP que sean consultables por humanos y máquinas

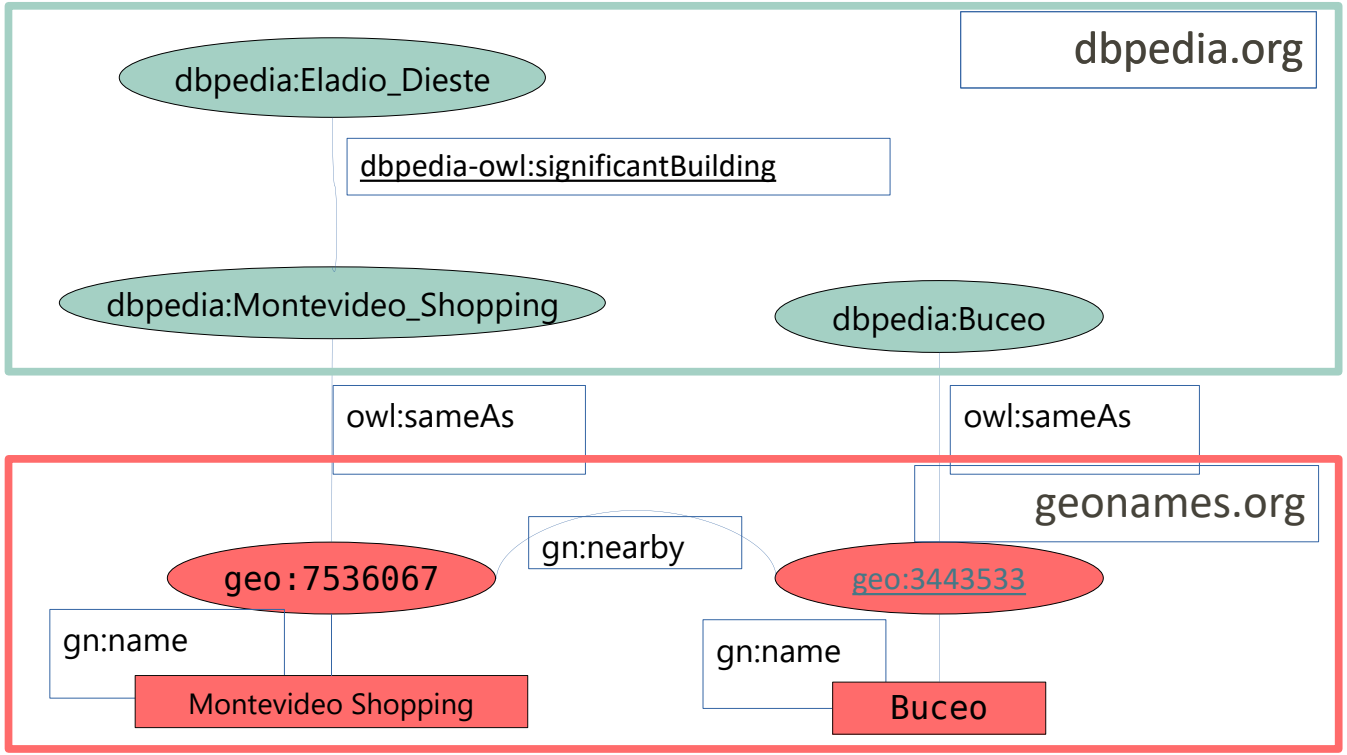
3) Proveer información útil acerca de cada URI en RDF

4) Crear links entre URIs

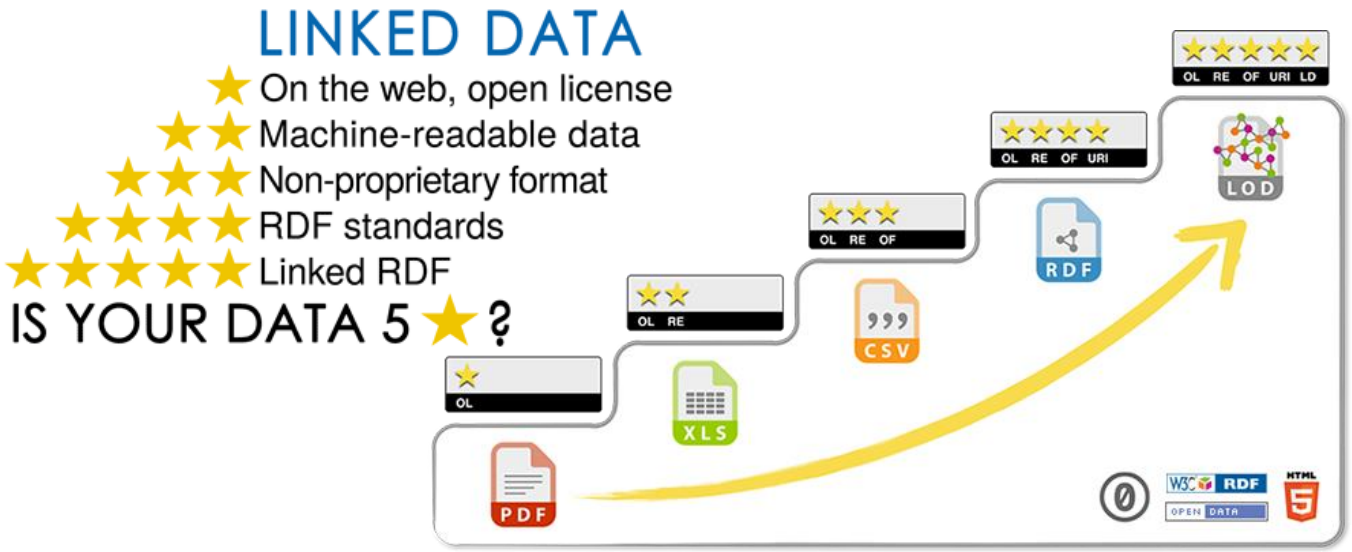
Tim Berners-Lee – Linked Data (2006)
http://www.w3.org/DesignIssues/LinkedData

#TED Talk Tim Berners-Lee on the next Web (2009)*http://www.ted.com/talks/tim_berners_lee_on_the_next_web.html*

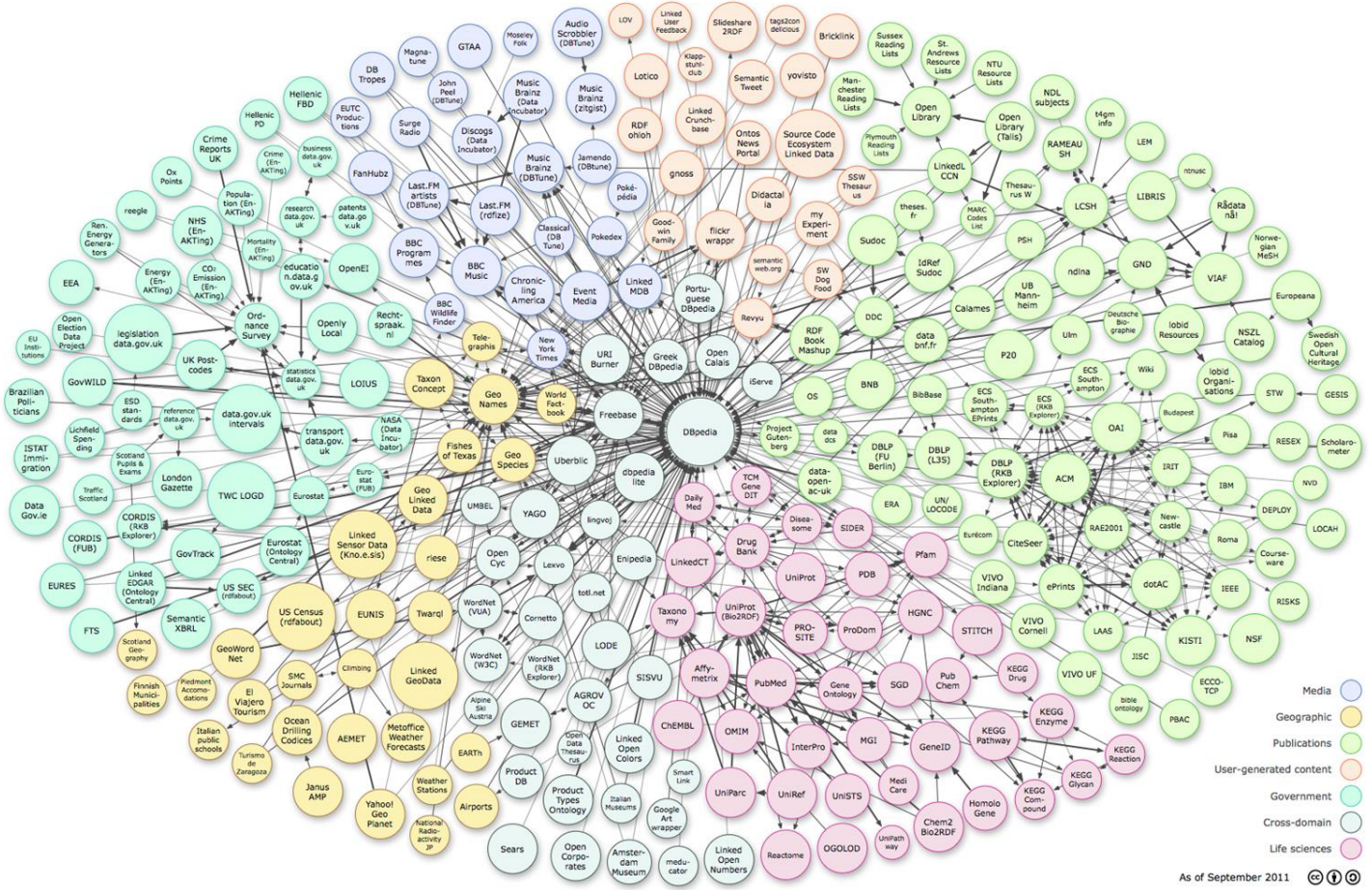
```
@prefix gn: http://www.geonames.org/ontology
@prefix dbpedia: http://dbpedia.org/resource/
@prefix dbpedia-owl: http://dbpedia.org/ontology/
@prefix geo:http://sws.geonames.org/
@prefix owl:http://www.w3.org/2002/07/owl#
```



Linked Data + Open Data = LOD



http://5stardata.info/en/

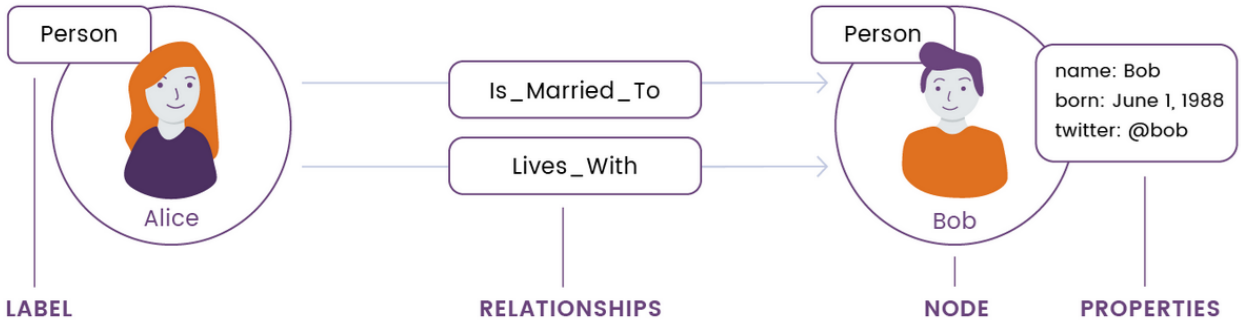


http://lod-cloud.net/ *2011*

El proceso de diseño

Consiste en decidir como representar a los elementos de la realidad en términos de nodos, etiquetas, relaciones y propiedades.

Elementos del PGM



Las etiquetas representan los tipos o roles de los nodos

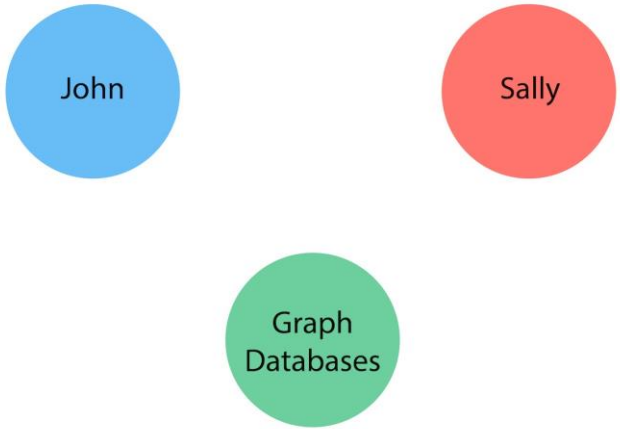
Las propiedades son parejas (clave,valor) y pueden estar asociadas a nodos y a relaciones (aristas).

Guías de diseño (1)

Nodos: usualmente representan **entidades** de la realidad.

Se pueden identificar nodos para el modelo de grafos a partir de sustantivos del dominio (ej: una persona, una empresa, etc.)

Two people, John and Sally, are friends. Both John and Sally have read the book, Graph Databases.



Fuente: https://neo4j.com/developer/guide-data-modeling/

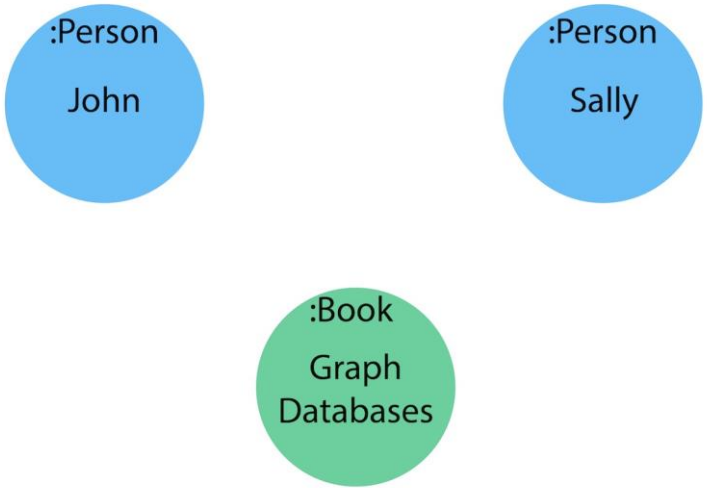
Guías de diseño (2)

Etiquetas: se usan para agrupar nodos en conjuntos.

Dan una noción “liviana” de tipos.

Son opcionales, y cada nodo puede tener más de una etiqueta.

Two people, John and Sally, are friends. Both John and Sally have read the book, Graph Databases.



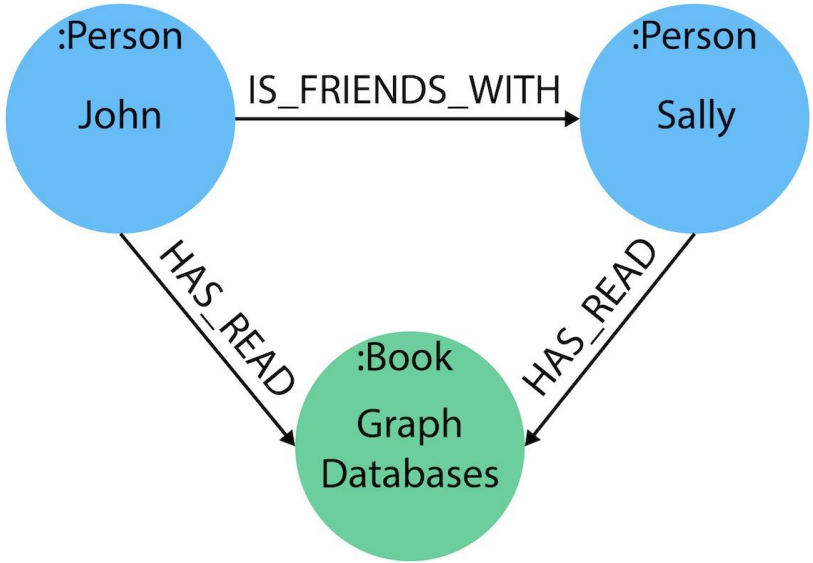
Fuente: https://neo4j.com/developer/guide-data-modeling/

Guías de diseño (3)

Relaciones: vinculan entidades (nodos) y tienen una dirección.

Se pueden identificar relaciones para el modelo de grafos a partir de verbos usados en el dominio
(ej: dirige, conoce, lee, etc.)

Two people, John and Sally, are friends. Both John and Sally have read the book, Graph Databases.



Fuente: https://neo4j.com/developer/guide-data-modeling/

Guías de diseño (4)

Propiedades: permiten almacenar datos sobre las relaciones y los nodos.

Para identificarlos se puede partir de los atributos relevantes identificados en el dominio, y en las preguntas que se quieren responder

Algunas preguntas posibles

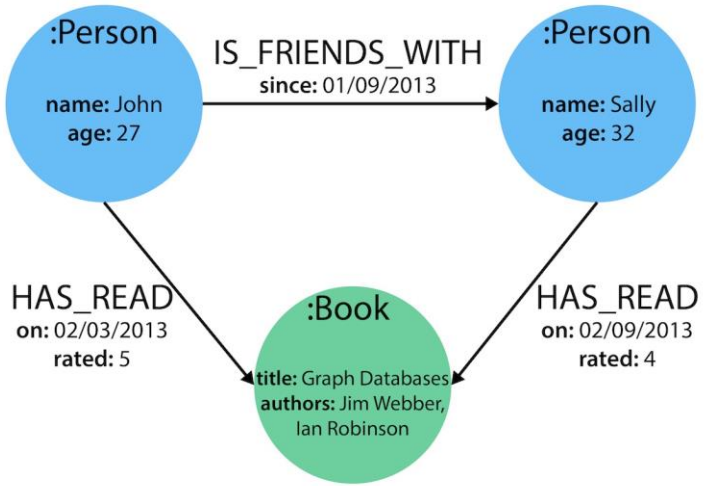
Cuánto hace que Sally y John s

Cuál es la calificación promedio

Quién es el autor del libro Grap

Cuál es la edad de Sally? Y la de

Quién leyó el libro Graph Datab



Fuente: https://neo4j.com/developer/guide-data-modeling/

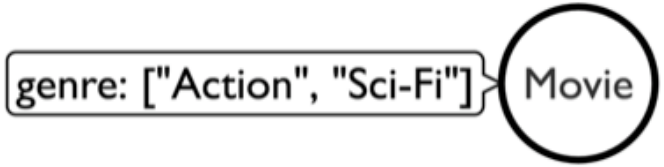
Decisiones de diseño

Más allá de las recomendaciones generales algunos conceptos de la realidad tienen más de una representación posible.

¿Cuál elegir? Depende ...

Veremos a continuación algunas variantes y discusiones

Propiedades vs. Relaciones



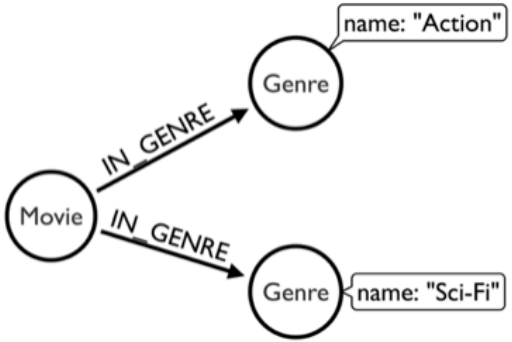
Devolver todos los géneros de una película

```
MATCH (m:Movie {title:"The Matrix"})
RETURN m.genre;
```

Devolver todas las películas que comparten géneros

```
MATCH (m1:Movie), (m2:Movie)
WHERE any(x IN m1.genre WHERE x IN m2.genre)
AND m1 <> m2
RETURN m1, m2;
```

Propiedades vs. Relaciones (2)



Devolver todos los géneros de una película

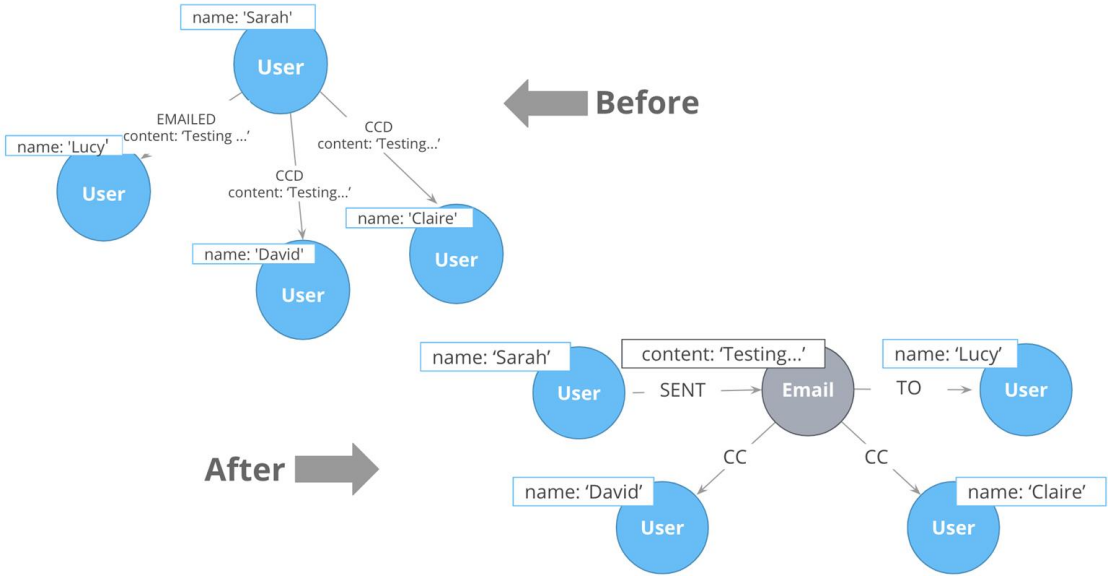
```
MATCH (m:Movie {title:"The Matrix"}),
      (m)-[:IN_GENRE]->(g:Genre)
RETURN g.name;
```

Devolver todas las películas que comparten géneros

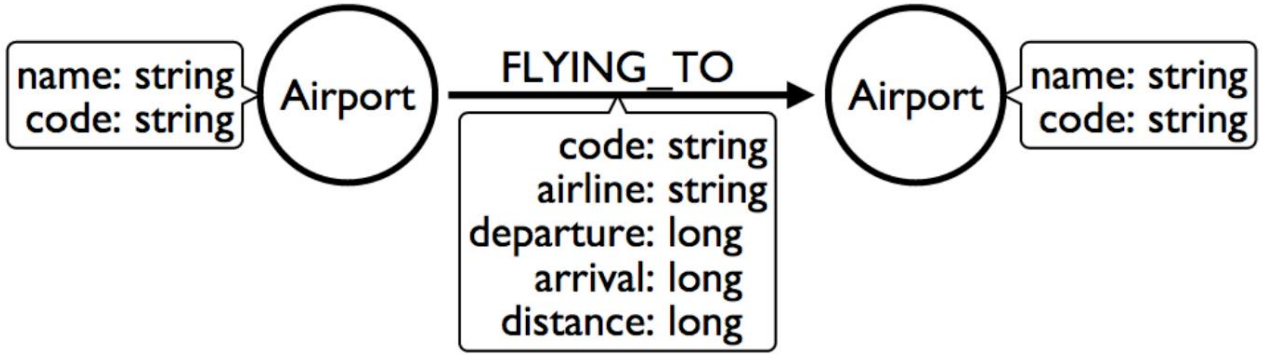
```
MATCH (m1:Movie)-[:IN_GENRE]->(g:Genre),
      (m2:Movie)-[:IN_GENRE]->(g)
RETURN m1, m2, g
```

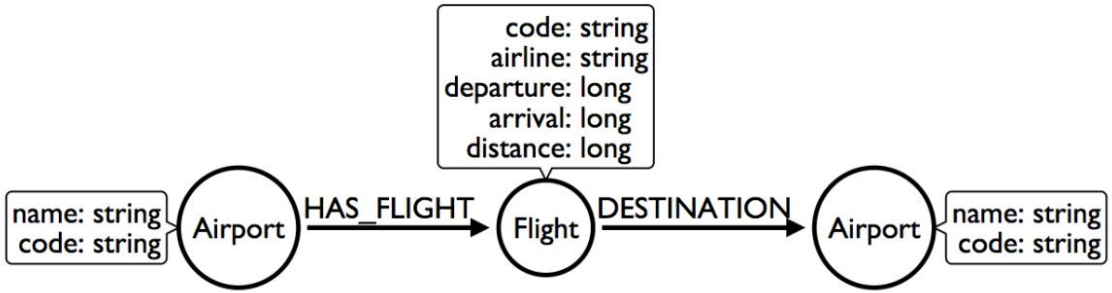
Hiper aristas o nodos intermedios

Permiten representar relaciones entre más de dos entidades



Versionado y datos dependientes del tiempo

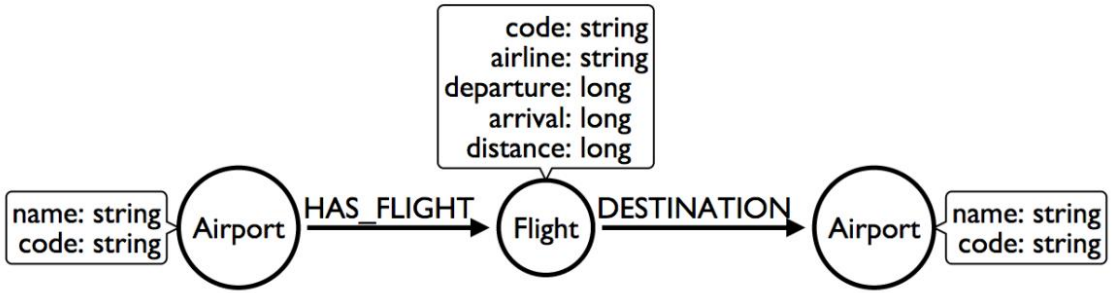




https://maxdemarzi.com/2015/08/26/modeling-airline-flights-in-neo4j/

Supongamos que se quiere usar el grafo en una aplicación para búsqueda y reserva de vuelos

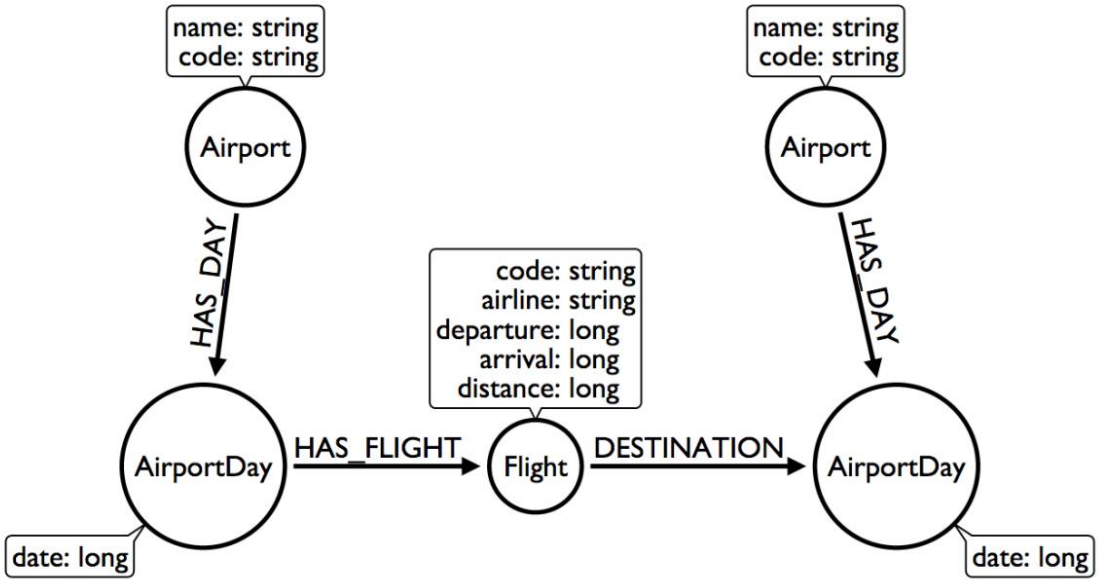
OBJETIVO: poder reducir rápidamente el subgrafo en que hago las búsquedas



Sabemos el origen y el destino que interesa, y también la fecha del vuelo

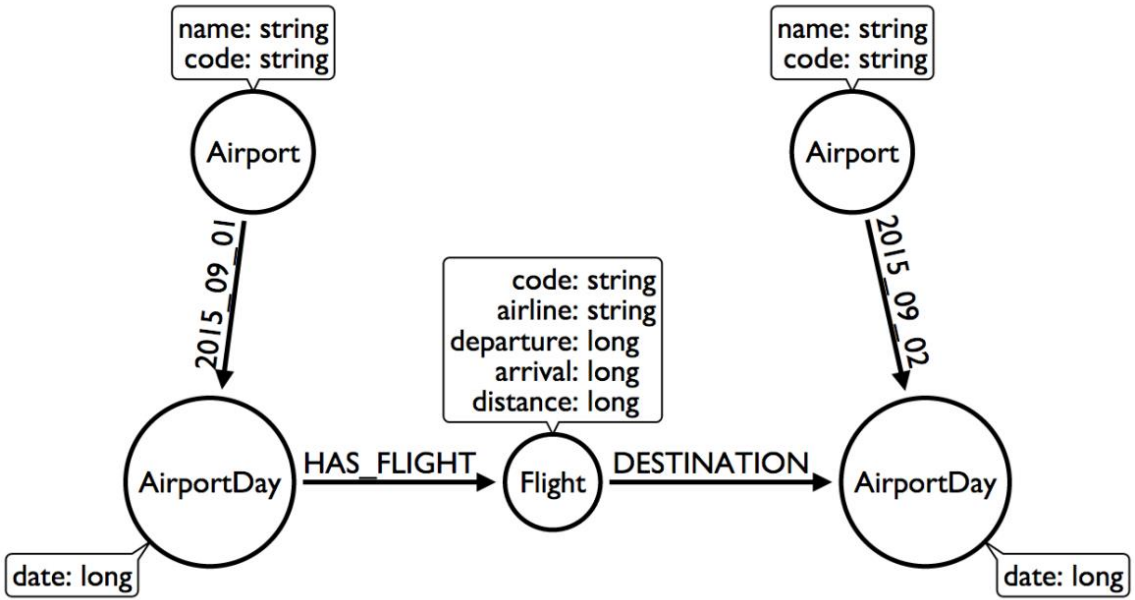
https://maxdemarzi.com/2015/08/26/modeling-airline-flights-in-neo4j/

Sabemos el origen y el destino que interesa, y también la fecha del vuelo



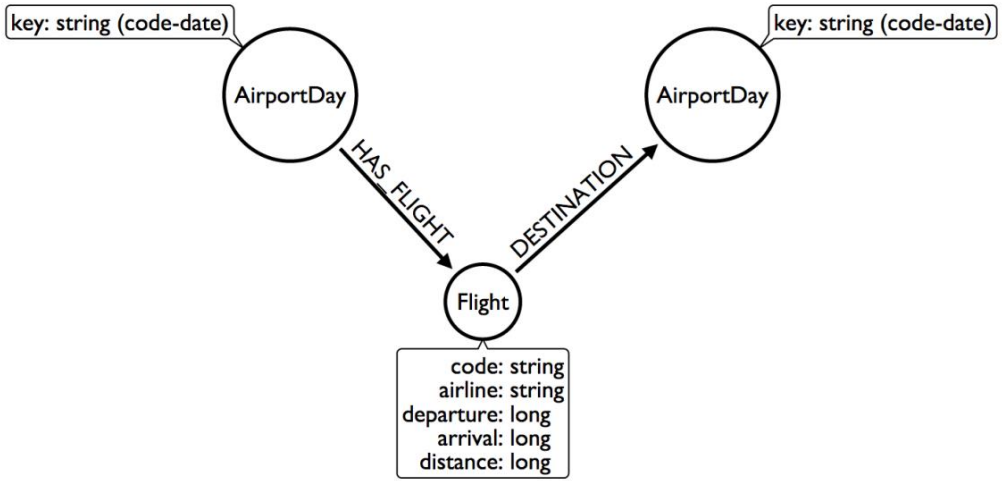
https://maxdemarzi.com/2015/08/26/modeling-airline-flights-in-neo4j/

Sabemos el origen y el destino que interesa, y también la fecha del vuelo



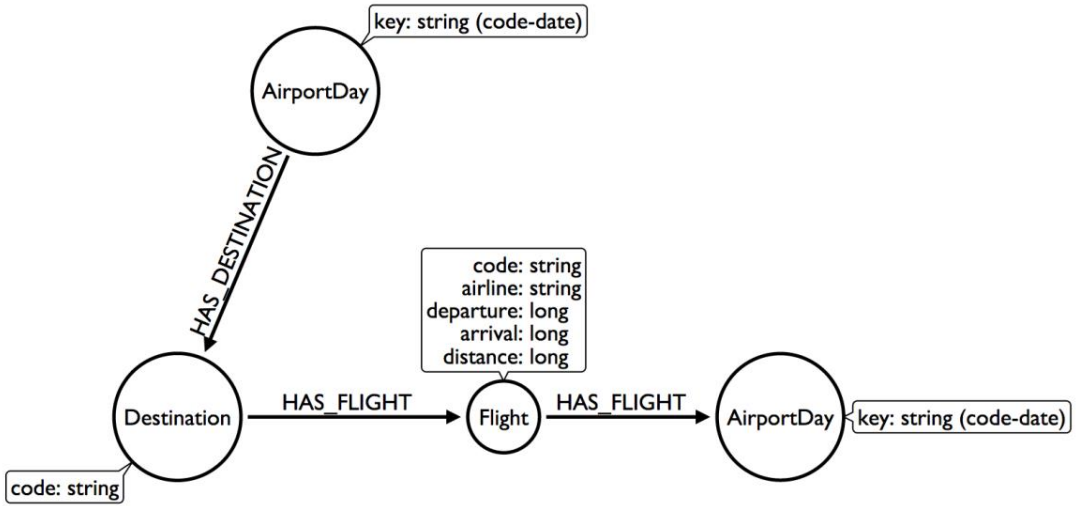
https://maxdemarzi.com/2015/08/26/modeling-airline-flights-in-neo4j/

Podemos mejorar el acceso al AirportDay combinando el código del aeropuerto y la fecha (epoch)



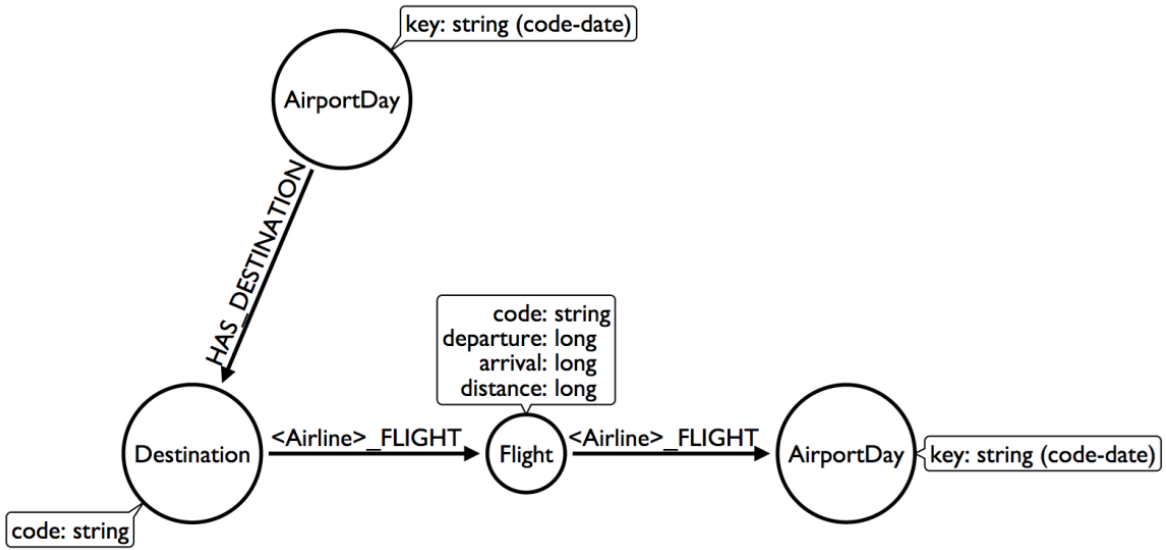
https://maxdemarzi.com/2015/08/26/modeling-airline-flights-in-neo4j/

Hay destinos que son muy populares (muchos vuelos).
Podríamos mejorar aún más el modelo



https://maxdemarzi.com/2015/08/26/modeling-airline-flights-in-neo4j/

Y si además queremos poder filtrar rápidamente por qué aerolínea opera el vuelo podríamos mejorar aún más el modelo



https://maxdemarzi.com/2015/08/26/modeling-airline-flights-in-neo4j/

¿qué pasa si ya tengo un diseño relacional?

Se puede comenzar aplicando la estrategia de mapeo directo:

- Entidades a nodos
- Relaciones binarias a aristas
- Nodos intermedios para relaciones de 3 o más

Bibliografía

Knowledge Graphs, Aidan Hogan, Eva Blomqvist, Michael Cochez, Claudia d'Amato, Gerard de Melo, Claudio Gutierrez, Sabrina Kirrane, José Emilio Labra Gayo, Roberto Navigli, Sebastian Neumaier, Axel-Cyrille Ngonga Ngomo, Axel Polleres, Sabbir M. Rashid, Anisa Rula, Lukas Schmelzeisen, Juan Sequeda, Steffen Staab, Antoine Zimmermann (2021) Synthesis Lectures on Data, Semantics, and Knowledge, No. 22, 1–237, DOI: 10.2200/S01125ED1V01Y202109DSK022, Springer. https://kgbook.org/

Next Generation Databases: NoSQLand Big Data, Guy Harrison (2016), Apress. https://link-springer-com.proxy.timbo.org.uy:88/chapter/10.1007/978-1-4842-1329-2_3

MongoDB: https://www.mongodb.com/

Cypher: https://neo4j.com/developer/cypher-query-language/