Diseño de bases de datos de documentos

Bases de Datos No Relacionales

Instituto de Computación, FING, UdelaR – 2023

Recapitulemos...

BD relacional

BD documental

Tabla	Colección
Fila	Documento
Columna	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

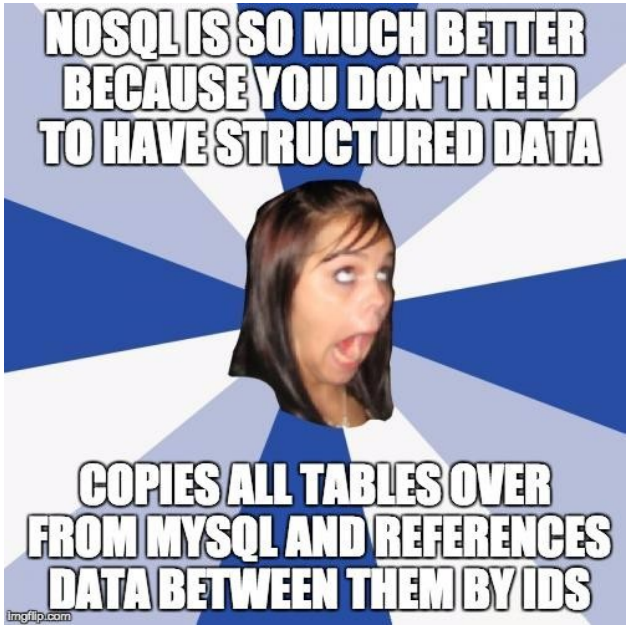
La ausencia de un esquema fijo (*schema-less*) no quiere decir que no exista una etapa de diseño y modelado

Diseño de BDs relacionales

- Hay heurísticas que permiten obtener un diseño a partir del modelo conceptual, garantizando cierta calidad del diseño obtenido
  - Pasaje de MER a ER (ver material)
- Esta calidad es independiente de las consultas a realizar.
- Las Formas Normales garantizan propiedades para todas las instancias

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",
  "description":"Actor, Battle of the Bastards"
}
```

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.

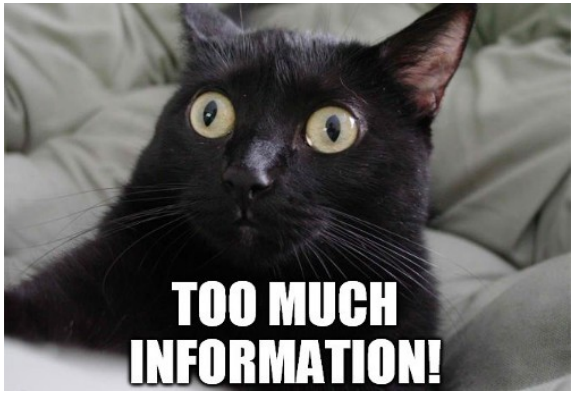
LIMITACIONES

- 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)

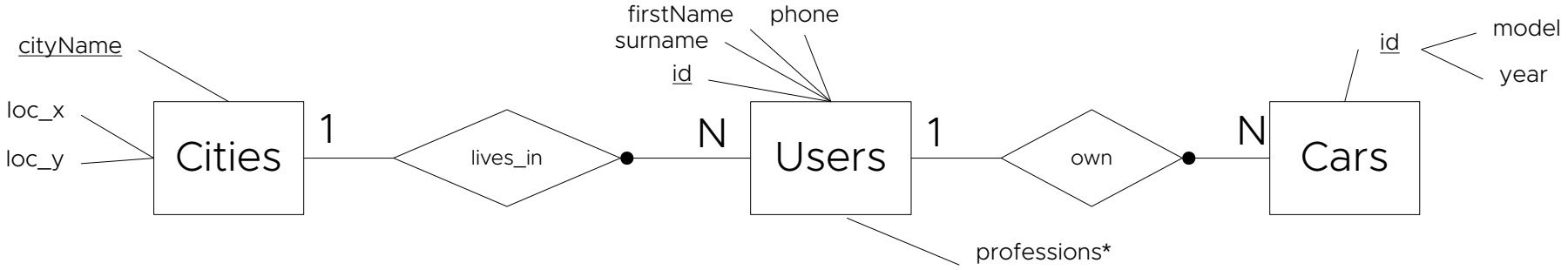
Estos aspectos condicionan los diseños posibles



¿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

Tabla Cities

cityName	loc_x	loc_y
London	45.123	47.232

Tabla Users

id	firstName	surname	phone	city
1	Paul	Miller	5111111	London

Tabla User_professions

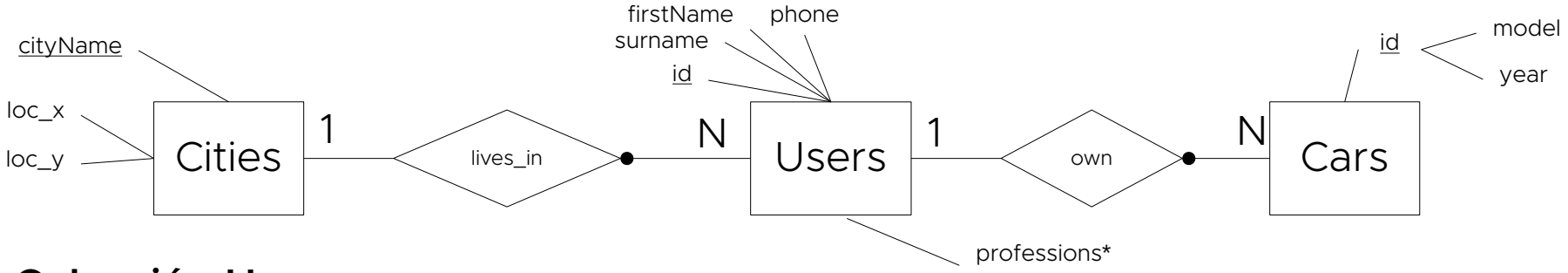
userId	profession
1	banking
1	finance
1	trader

Tabla Cars

userId	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": [45.123, 47.232],
   "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

Recomendaciones de diseño (1)

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

- Relaciones 1:N :
- 1 a pocos – Preferir documentos embebidos
- 1 a algunos - Preferir documentos embebidos
- 1 a muchísimos - Preferir documentos referenciados

- Relaciones N:N - Preferir documentos referenciados

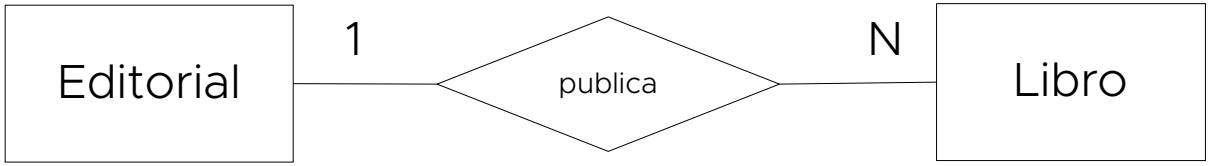
Recomendaciones de diseño (2)

- *Rule 1: Favor embedding unless there is a compelling reason not to.*
- *Rule 2: Needing to access an object on its own is a compelling reason not to embed it.*
- *Rule 3: Avoid joins/lookups if possible, but don't be afraid if they can provide a better schema design.*

MongoDB Schema Design Best Practices, MongoDB 2022

¿Dónde almaceno las relaciones 1:N?

Cuando usamos referencias en relaciones 1:N :
¿de qué lado de la relación almaceno la referencia?



Depende de la cantidad y sobre todo de cuan dinámicas sean estas relaciones

Si la cantidad de libros es acotada y cambia poco

```
{
    name: "Apress",
    founded: 1999,
    location: "US",
    books: [123456, 456789, ...]
}
```

```
{
    _id: 456789,
    title: "MongoDB Recipes",
    author: [ "Subhashini Chellappan"],
    published_date: ISODate("2018-11-30"),
    pages: 120,
    language: "English"
}
```

Pero si cambia mucho y no es acotable a priori

```
{
    _id:"Apress",
    name: "Apress",
    founded: 1999,
    location: "US"
}
```

```
{
    _id: 456789,
    title: "MongoDB Recipes",
    author: [ "Subhashini Chellappan"],
    published_date: ISODate("2018-11-30"),
    pages: 120,
    language: "English",
    publisher_id: "Apress"
}
```

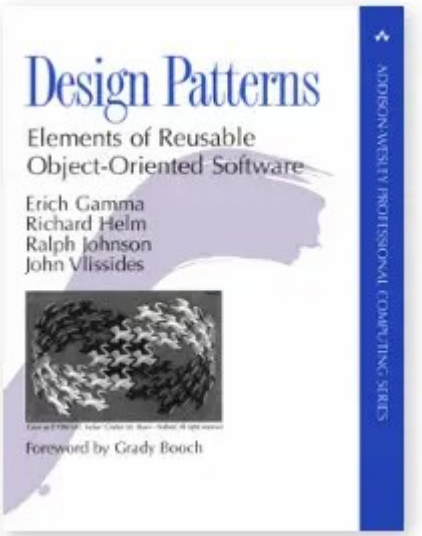
Recomendaciones de diseño (3)

- *Rule 4: Arrays should not grow without bound. If there are more than a couple of hundred documents on the "many" side, don't embed them; if there are more than a few thousand documents on the "many" side, don't use an array of ObjectID references. High-cardinality arrays are a compelling reason not to embed.*

MongoDB Schema Design Best Practices, MongoDB 2022

Patrones de diseño

- Similares a los patrones de diseño de software ➡
- Resuelven algunas situaciones comunes
- Permiten hacer el proceso de diseño menos artesanal



Atención!

Típicamente buscan mejorar la eficiencia de consultas y por lo tanto pueden introducir...

- Duplicación de datos
- Datos desactualizados
- Problemas de integridad referencial

Duplicación de datos

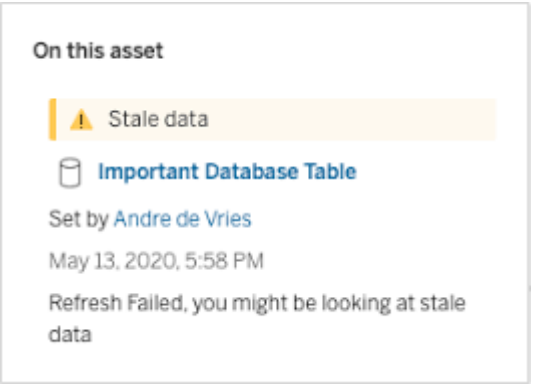
¿Cuándo?

- Al embeber datos en un documento para hacer el acceso más eficiente

¿Preocupación?

- Puede ser un desafío mantener la corrección y la consistencia

Datos desactualizados



¿Cuándo?

- Los datos cambian y el costo de actualizarlos permanentemente es demasiado alto

¿Preocupación?

- Calidad de datos y confiabilidad

Problemas de integridad referencial

¿Cuándo?

- Referencias a documentos que ya no existen (no hay FKs ni *cascade deletes*)

¿Preocupación?

- Calidad de datos y confiabilidad

Algunos patrones interesantes

- Attribute pattern
- Extended reference pattern
- Subset pattern
- Tree patterns
- Schema versioning pattern

La lista completa puede consultarse aquí.

Attribute pattern

Problema

- Muchos atributos similares
- Necesidad de buscar por varios atributos a la vez

Solución

- Transformar atributo/valor en:
  - atrA: *atributo*
  - atrB: *valor*
- *Ejemplo:*
  - *{“color”:”azul”, “tamaño”: grande}*
  - *{[{“k”:”color, “v”:”azul”}*
    {“k”:”tamaño, “v”:”grande”}]}

Casos de uso de ejemplo

```
{
  "title": "Dunkirk",
  ...
  "release_USA": "2017/07/23",
  "release_Mexico": "2017/08/01",
  "release_France": "2017/08/01",
  "release_Festival_San_Jose":
    "2017/07/22"
}
```

```
{
  "title:" "Dunkirk",
  ...
  "releases": [ {
    { "k": "release_USA",
      "v": "2017/07/23" },
    { "k": "release_Mexico",
      "v": "2017/08/01" },
    { "k": "release_France",
      "v": "2017/08/01" },
    { "k": "release_Festival_San_Jose",
      "v": "2017/07/22" }
  ]
}
```

```
db.movies.find({"releases.v":{$gte:"2017/07",$lt:"2017/08"}})
```

Beneficios y desventajas

- ✅ Más sencillo de indexar
- ✅ No es necesario conocer los nombres de los atributos a indexar

Extended reference pattern

Problema

- Frecuentemente realizo joins para recuperar algunos atributos

Solución

- Identificar atributos del lado del lookup
- Copiar estos atributos en el documento que hace la referencia

Casos de uso de ejemplo

- Catálogo de productos
- Aplicaciones móviles
- Análisis real-time

Beneficios y desventajas

✅ Lecturas más rápidas

✅ Reduce el número de joins y lookups

❌ Genera duplicación, q puede ser difícil de gestionar si los campos duplicados cambian mucho

Subset pattern

Problema

- Necesidad de reducir el tamaño del conjunto de documentos (demasiado grandes para trabajar en memoria)
- Hay atributos que se usan poco

Solución

- Partir la colección en dos:
  - Atributos más usados
  - Atributos menos usados
- Duplico parte de un conjunto en el documento

Casos de uso de ejemplo

- Listas de reviews de un producto
  - Guardo un subconjunto en el producto
- Listas de comentarios de películas
- Elenco de una película

Beneficios y desventajas

✅ El tamaño del conjunto de docs se reduce

✅ Menor costo de acceso a disco

❌ Más round trip al servidor

❌ Uso un poco más de espacio en disco

Tree patterns

Problema

- Representar información jerárquica

Solución

- Varias!
  - Referencia a los hijos
  - Referencia al padre
  - Arreglo de ancestros
  - Caminos materializados

Casos de uso de ejemplo

- Categorías de productos
- Organigramas

Beneficios y desventajas

✅ Referencia a los hijos: fácil navegar hacia abajo

✅ Referencia al padre: fácil navegar hacia arriba

✅ Arreglo de ancestros: navegar hacia arriba sin tener q recorrer todo

✅ Caminos materializados: expresiones regulares para buscar

Schema versioning pattern

Problema

- Gestionar cambios en el esquema reduciendo el downtime

Solución

- Cada documento indica la versión del esquema a la que adhiere
- La aplicación puede gestionar todas las versiones
- Elijo una estrategia para migrar los documentos

Casos de uso de ejemplo

- Cualquiera que precise hacer cambios

Beneficios y desventajas

✅ Evito el downtime

✅ Mantengo la migración bajo control

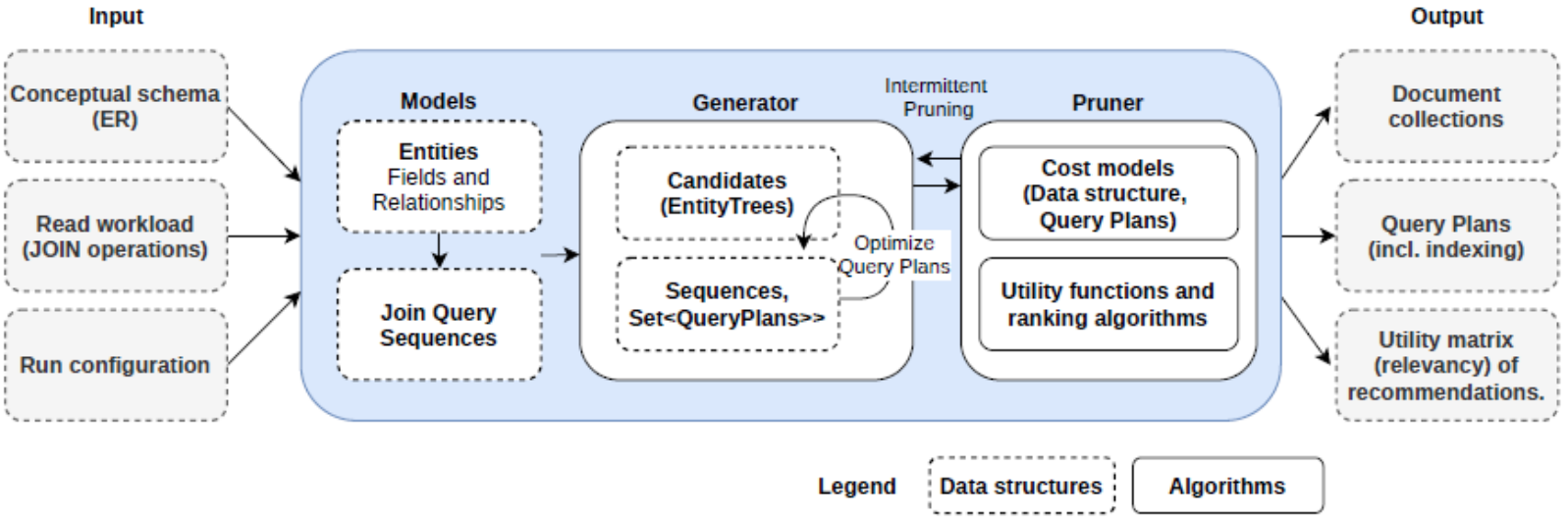
✅ Reducción de la deuda técnica

Metodologías emergentes

A Workload-driven Document Database Schema Recommender (DBSR)

Vincent Reniers(✉)[0000−0003−3895−702X], Dimitri Van Landuyt[0000−0001−6597−2271], Ansar Rafique[0000−0002−5881−7588], and Wouter Joosen[0000−0002−7710−5092]

imec-DistriNet, KU Leuven
Celestijnenlaan 200A, 3000 Leuven, Belgium
first.lastname@cs.kuleuven.be



Reniers, V., et al. (2020). A Workload-Driven Document Database Schema Recommender (DBSR). In ER 2020, Proceedings 39 (pp. 471-484). Springer.

Metodologías emergentes (2)

Knowledge and Information Systems
https://doi.org/10.1007/s10115-023-01828-3

REGULAR PAPER

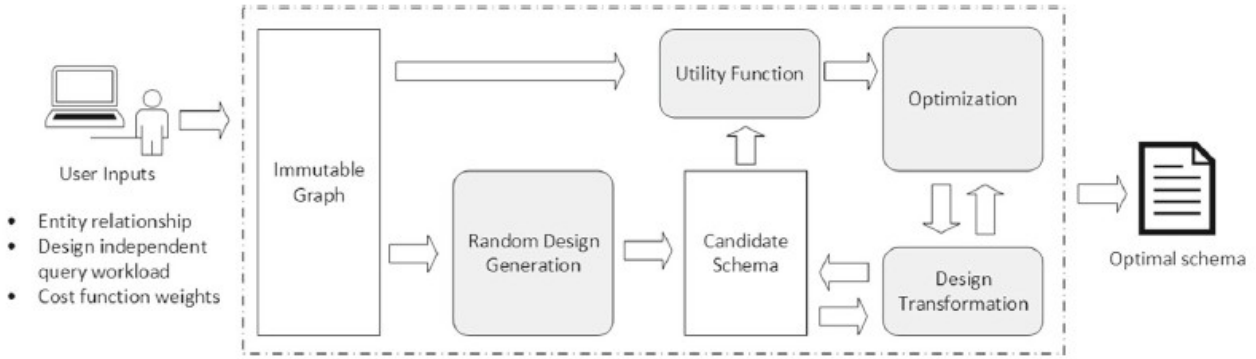
Automated database design for document stores with multicriteria optimization

Moditha Hewasinghage[1] · Sergi Nadal[1] · Alberto Abelló[1] · Esteban Zimányi[2]

Received: 28 September 2021 / Revised: 31 December 2022 / Accepted: 6 January 2023

Table 1 Overview of existing tools for schema generation

System	Data store	Workload	Optimization	Exploration
Chebotko et al. [24]	Col	Read	Rule-based	Low
Mortadelo [7]	Col/Doc	Read	Query cost/metamodel	Low
De Lima et al. [23]	Doc	Read	Query cost/rule-based	Low
NoSE [6]	Col	Read	Query cost	Low
DBSR [5]	Doc	Read	Query cost	Low
DocDesign 2.0	Doc	Read	Multicriteria	High



Hewasinghage, M., Nadal, S., Abelló, A., & Zimányi, E. (2023). Automated database design for document stores with multicriteria optimization. Knowledge and Information Systems, 1-34.

Referencias adicionales

- A Complete Methodology of Data Modeling for MongoDB, MongoDB

  https://www.youtube.com/watch?v=DUCvYbcgGsQ

- Advanced Schema Design Patterns, MongoDB

  https://www.youtube.com/watch?v=bxw1AkH2aM4&t=374s

- Documentación en el sitio de MongoDB
  https://docs.mongodb.com/manual/data-modeling/

- MongoDB Applied Design Patterns, Rick Copeland, Oreilly 2013 CASOS DE USO