
Base de datos no relacionales

Edición 2021

Implementación algoritmo R2G

Grupo 10:
Bruno De Simone
Matías Leal

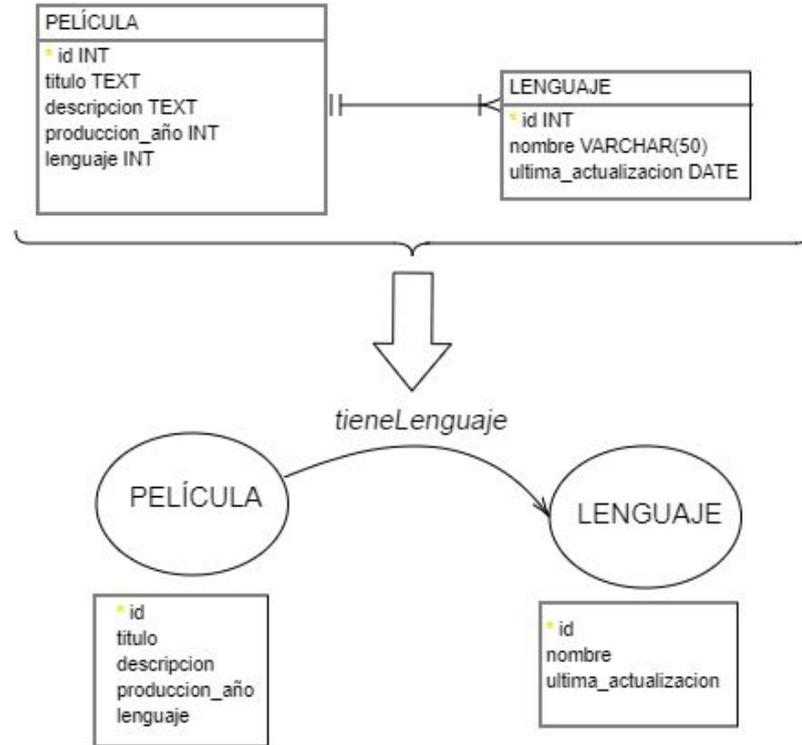
The Road So Far...

- Idea de implementar algo...
 - !!! Código en GitHub encontrado...
 - :(El código no funciona
 - El código ahora funciona! :D
 - Investigar el artículo de los Italianos... Diferencias?
 - Hora de aplicar cambios...
 - Cambios incompletos
 - Pero...
-

Algoritmo “Naïve” (GitHub)

- Es una estrategia de “mapeo” directo.
 - Básicamente se traduce el modelo Entidad-Relación a un modelo de grafo (de manera coherente) y de esta forma:
 - Entidad  Vértice
 - Relación  Arista
 - Pros:
 - Rápido
 - Pocas reglas en el mapeo
 - Contras:
 - El grafo resultante puede tener un overhead.
-

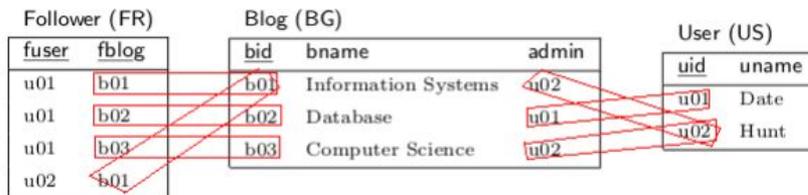
Algoritmo "Naïve" - Ejemplo



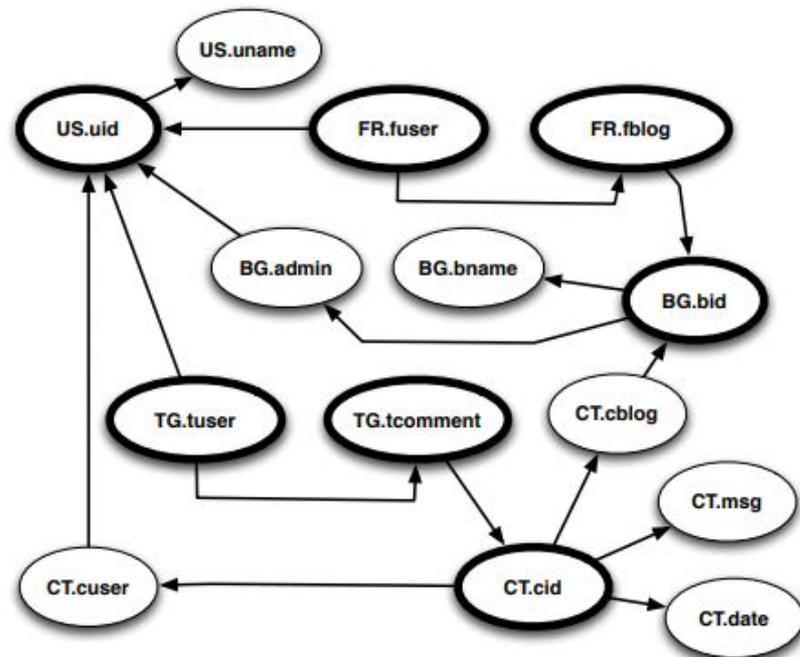
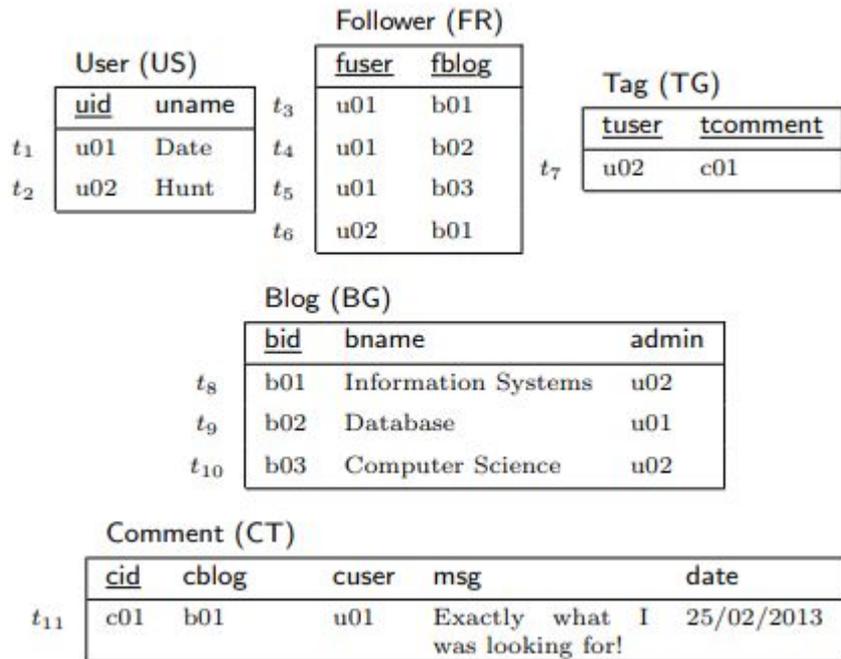
Algoritmo R2G (“paper italianos”)

- Es un algoritmo que le incluye el mapeo del modelo relacional al modelo de grafos, una migración automática.
 - Asegura que toda consulta que se pueda escribir en el modelo relacional, se plasma en una recorrida del grafo generado.
 - Minimiza la cantidad de accesos necesarios para resolver consultas.
 - Sin ataduras a ningún GDBMS en particular.
-

- **Joinable** tuples $t_1 \in R_1$ and $t_2 \in R_2$:
 - there is a foreign key constraint between $R_1.A$ and $R_2.B$ and $t_1[A] = t_2[B]$.
- **Unifiability** of data values $t_1[A]$ and $t_2[B]$:
 - (i) $t_1=t_2$ and both A and B do not belong to a multi-attribute key;
 - (ii) t_1 and t_2 are joinable and A belongs to a multi-attribute key;
 - (iii) t_1 and t_2 are joinable, A and B do not belong to a multi-attribute key and there is no other tuple t_3 that is joinable with t_2 .



Algoritmo R2G (full scheme paths)



Algoritmo R2G

Algorithm 1: Create a graph database g

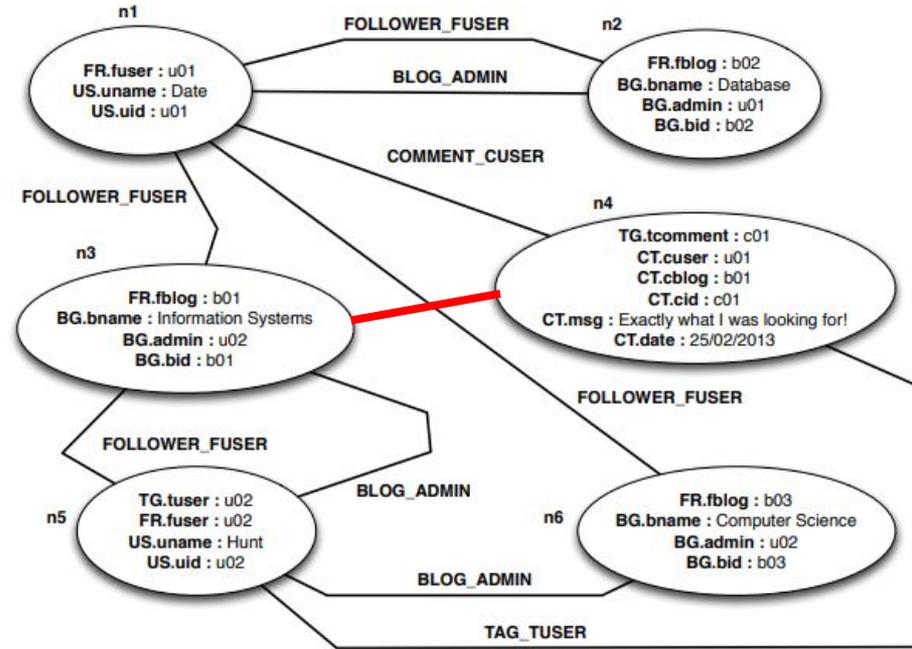
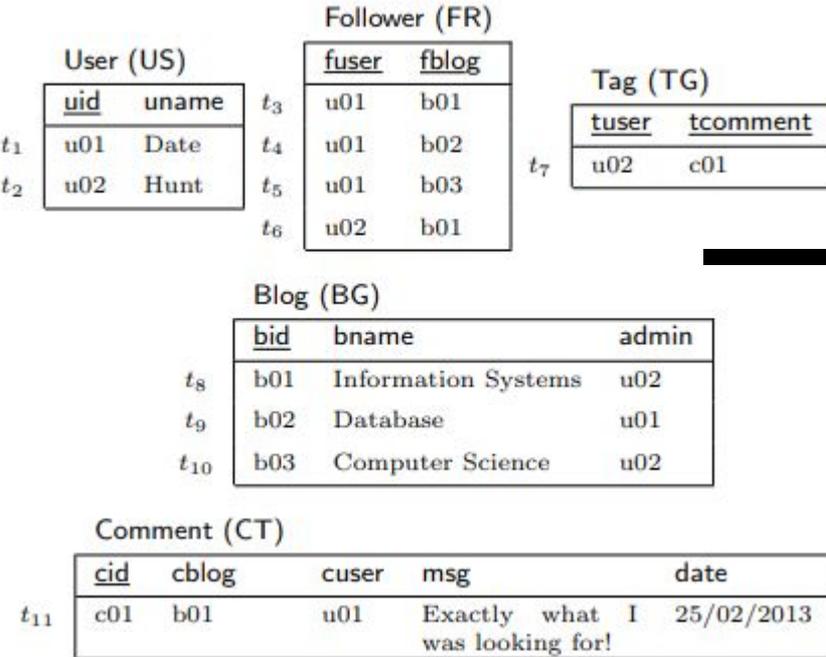
Input : A relational database r , a set SP of full schema paths

Output: A graph database g

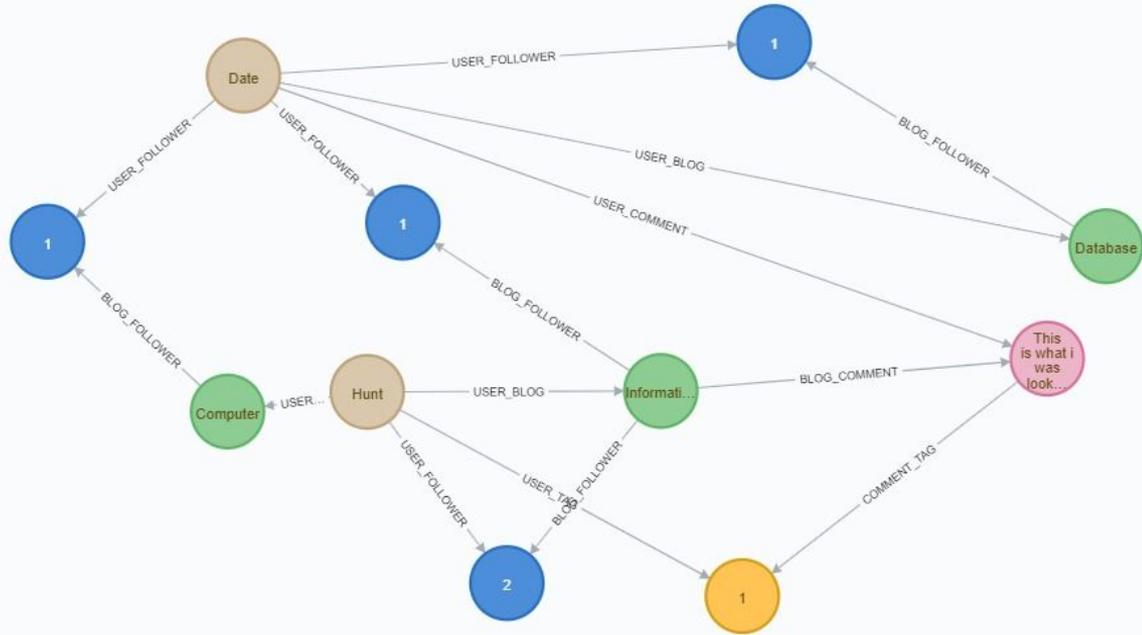
```
1  $VS \leftarrow \emptyset$ ;  
2  $g \leftarrow (\emptyset, \emptyset)$ ;  
3 foreach  $sp \in SP$  do  
4     foreach  $A_i \in sp$  do  
5         switch  $\text{cond}(sp, A_i, VS)$  do  
6             case 1  $\text{NewNode}(A_i, r, g)$ ;  
7             case 2  $\text{NewProperty}(A_i, r, g)$ ;  
8             case 3  $\text{NewProperty}(A_i, sp, r, g)$ ;  
9             case 4  $\text{NewNodeEdge}(A_i, sp, r, g)$ ;  
10            case 5  $\text{NewEdge}(A_i, sp, r, g)$ ;  
11          $VS \leftarrow VS \cup \{A_i\}$ ;  
12     return  $g$ ;
```

```
for _, path := range sp {  
    if len(path) > 0 {  
        for i, node := range path {  
  
            aux := strings.Split(node, ".")  
  
            //logga(node + ": " + strconv.Itoa(cond(path, i, &vs,  
  
            switch cond(path, i, &vs, &n2n, aux[2] == "ishub") {  
            case 1:  
                NewNode(db, []string{node})  
            case 2:  
               NewProperty(db, []string{path[i+1], node})  
            case 3:  
               NewProperty2(db, []string{path[i-1], node})  
            case 6:  
               NewProperty(db, []string{path[i-1], node})  
            case 4:  
               NewNodeEdge(db, []string{path[i-1], node})  
            case 5:  
               NewEdge(db, []string{path[i-1], node}, &n2n)  
            }  
        }  
    }  
}
```

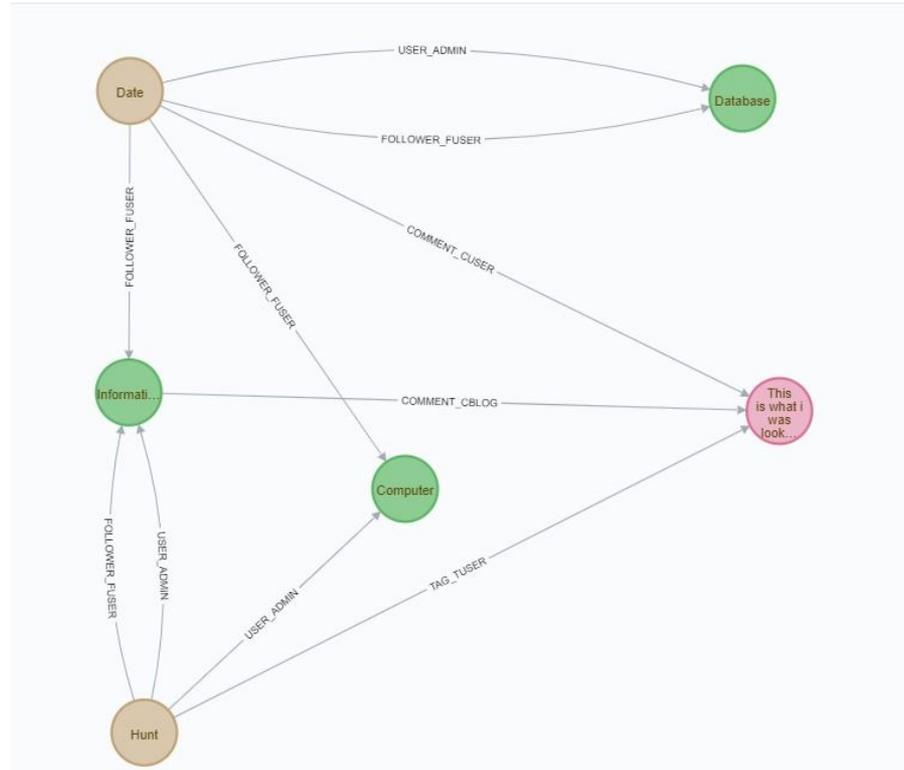
Algoritmo R2G (Ejemplo)



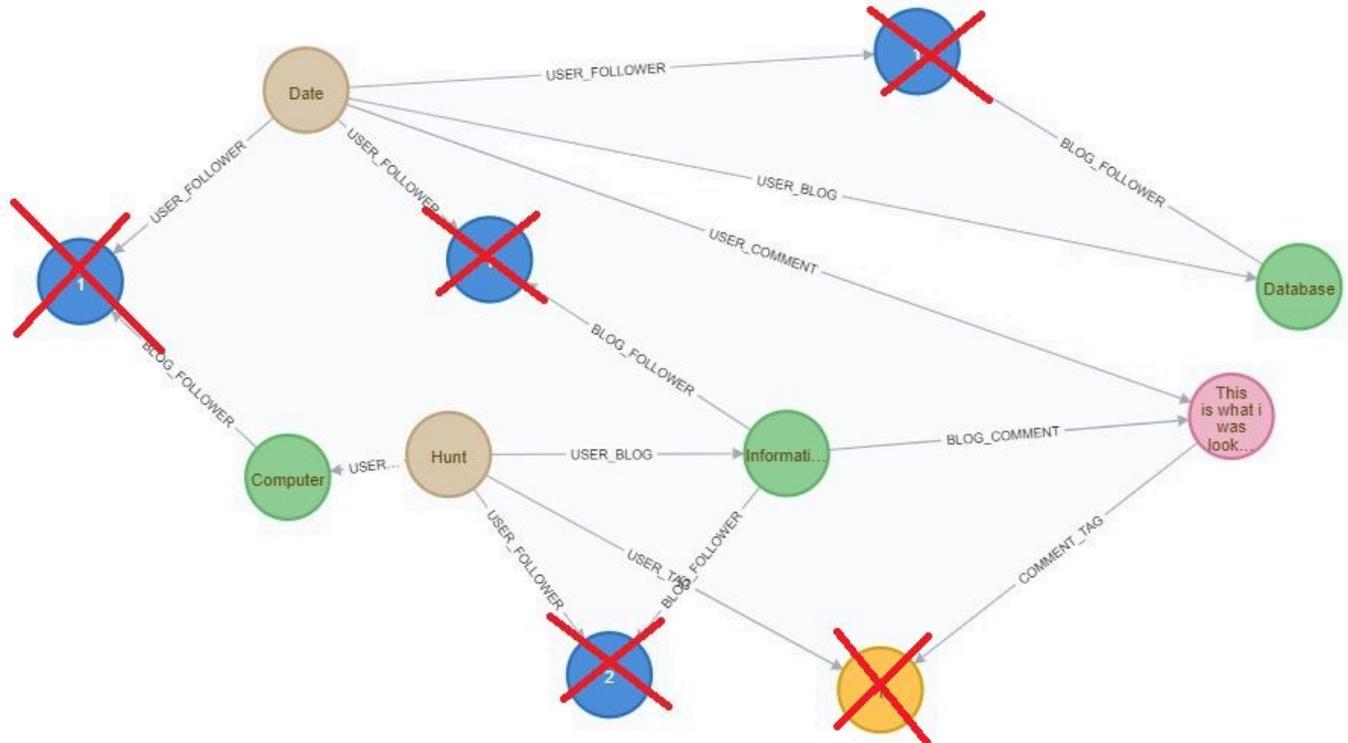
R2G vs Naïve (Naïve)



R2G vs Naïve (R2G)



R2G vs Naïve (diferencias)



R2G: Benchmark

Se utilizó una base de datos con la misma estructura del paper estudiado

Plataforma de cómputo:

CPU Ryzen 1700X, 12Gb de RAM

R2G: Resultados

- Nombre de usuario y blogs que sigue:
 - MYSQL: 0.041s
 - NEO4j-R2G: 0.068s
 - Nombre de usuario y mensaje de comentario donde fueron taggeados:
 - MYSQL: 3.760s
 - NEO4j-R2G: 1.554s
 - Comentarios en los que "Omari Harris" fue taggeado:
 - MYSQL: 0.056s
 - NEO4j-R2G: 0.583s
-

R2G: Conclusiones

- “Revivir un muerto” es algo engorroso (código GitHub)
 - El mapeo Naïve no es nada atractivo para elegir en caso de que se tenga una base de datos muy grande.
 - Ventana abierta a terminar la implementación del código de R2G.
 - Los nodos no tienen identificación.
-

Trabajo a futuro

- Limpiar código del repo
 - Implementación completa del algoritmo
 - Agregar la funcionalidad de detectar n^2n (fácil pero no hubo tiempo)
-

Programa en acción

