An abstract graphic on the left side of the slide, featuring a vibrant red background with flowing, translucent green and yellow shapes that create a sense of movement and depth.

BASES DE DATOS BASADAS EN GRAFOS

Ejemplos con Neo4J

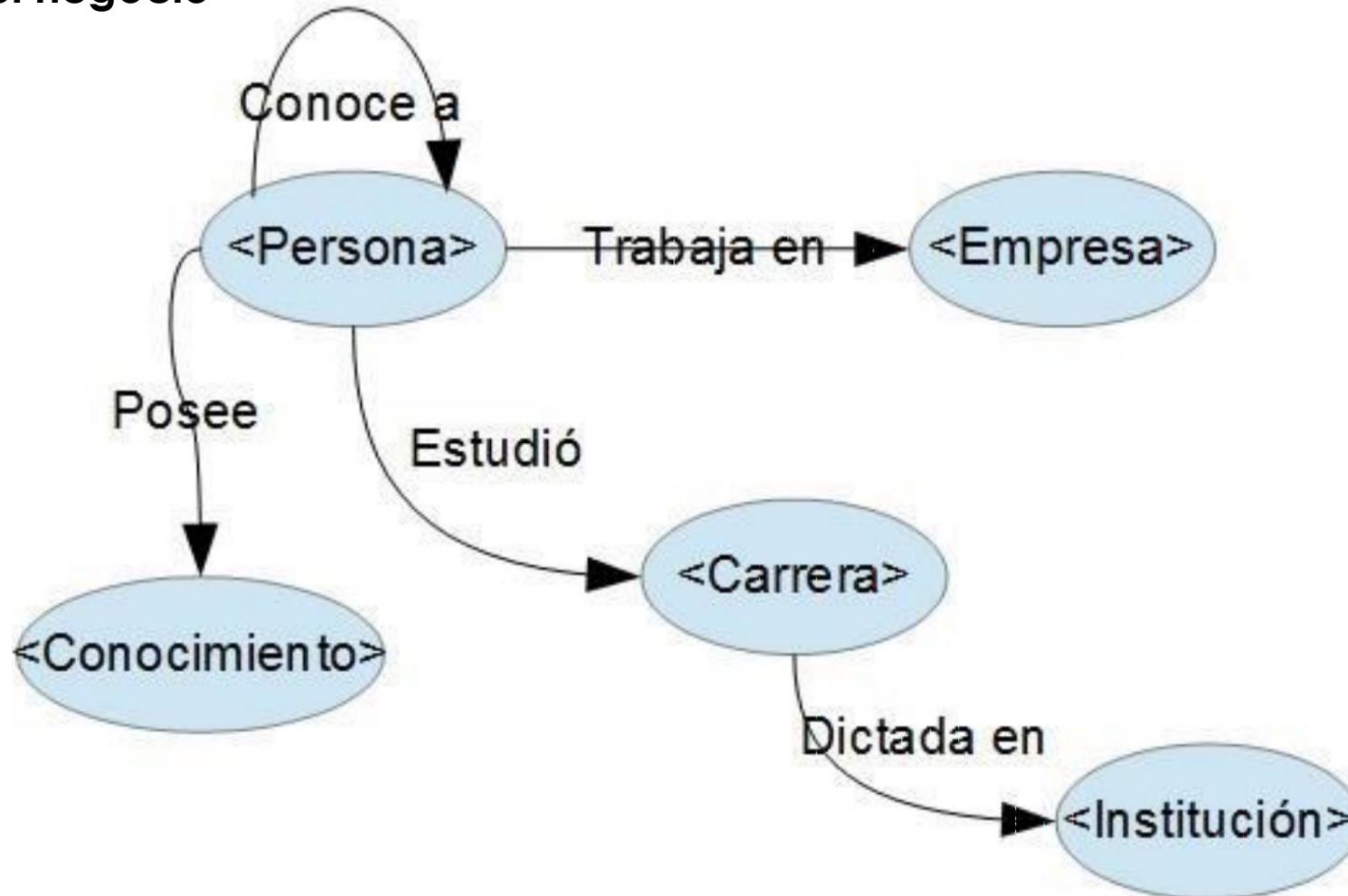


GRAPH BASED

EMPEZAMOS CON UN
CASO PRÁCTICO

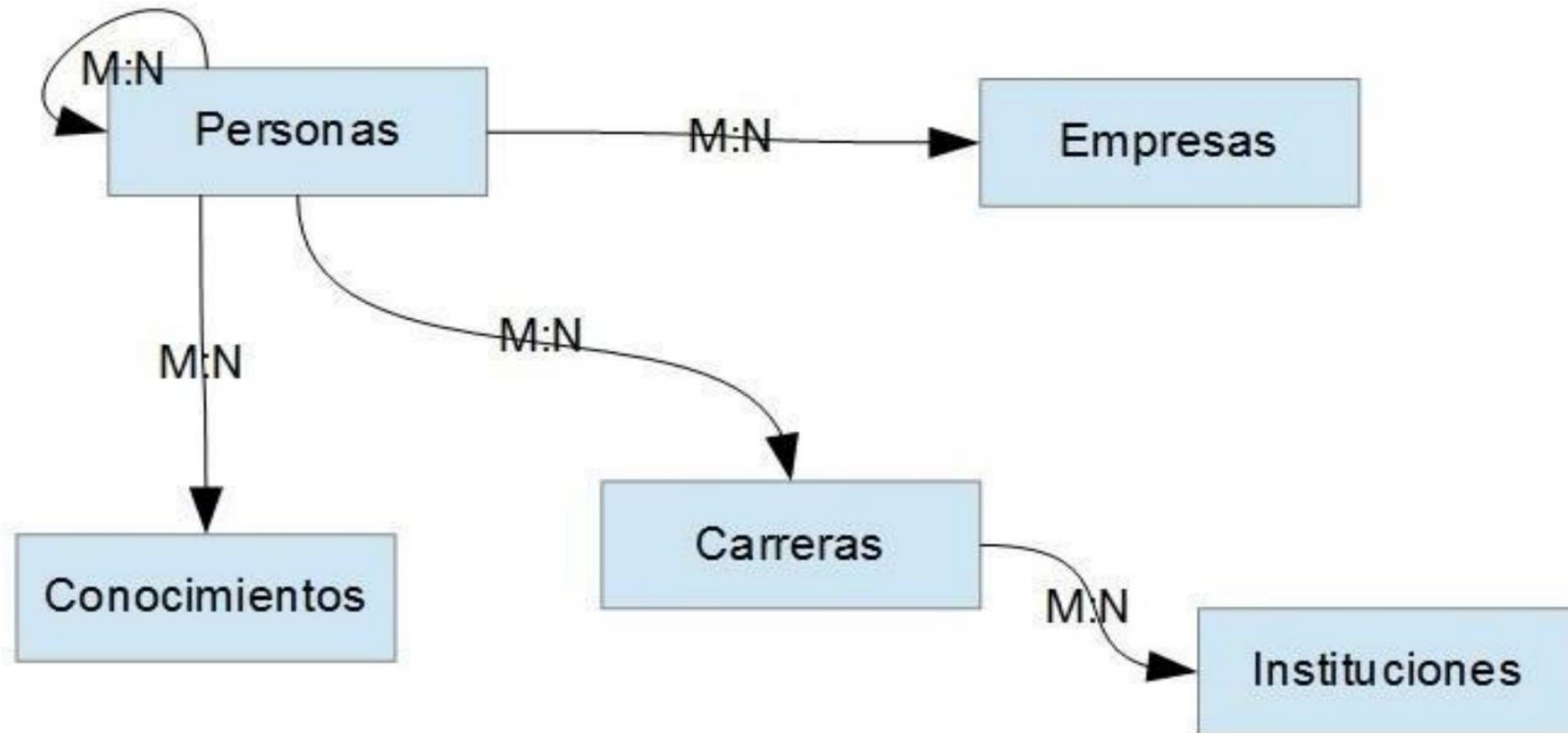
CASO PRÁCTICO: RED SOCIAL PROFESIONAL

Dominio del negocio



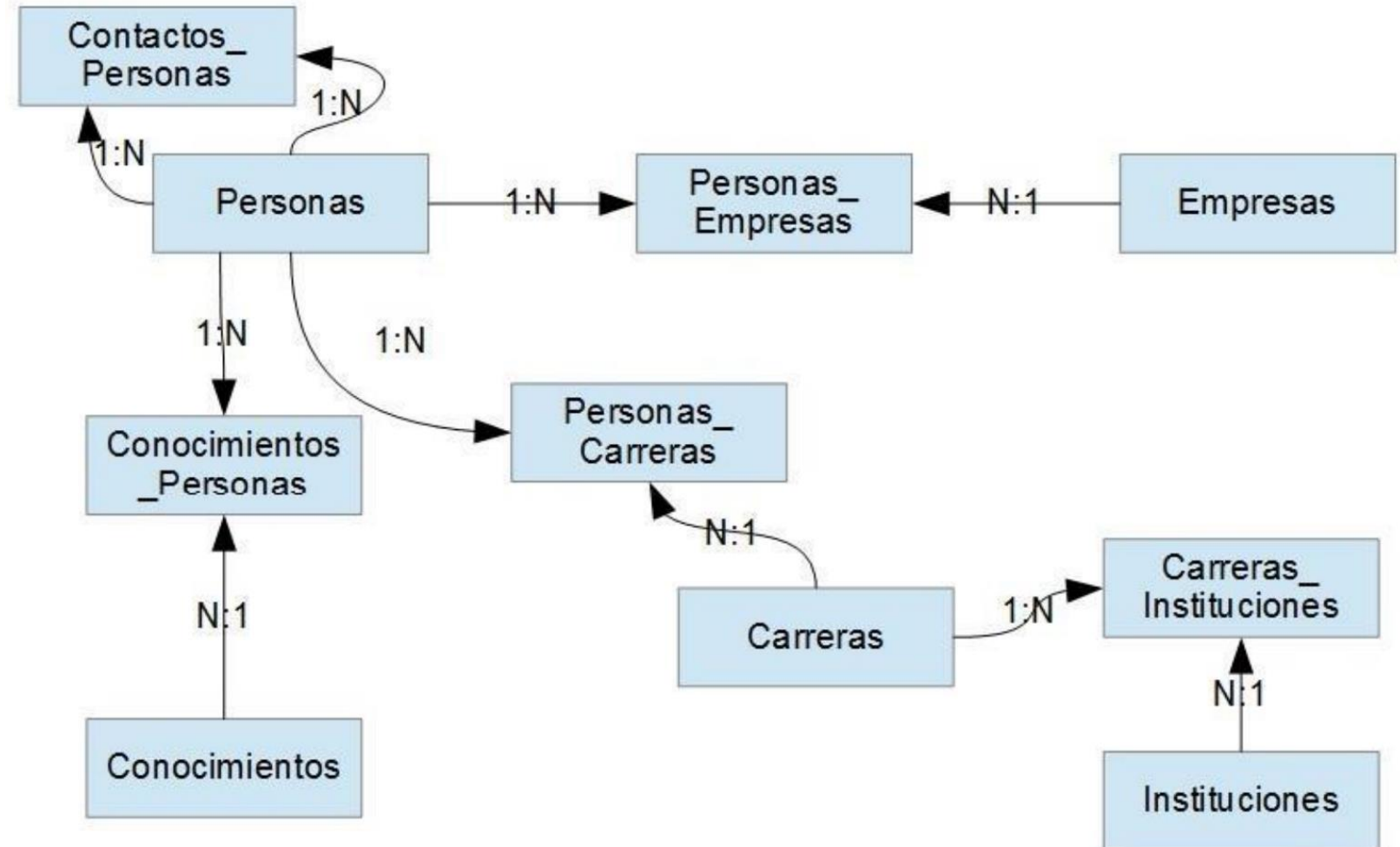
CASO PRÁCTICO: RED SOCIAL PROFESIONAL

Diseño Lógico Relacional



CASO PRÁCTICO: RED SOCIAL PROFESIONAL

Diseño Físico Relacional



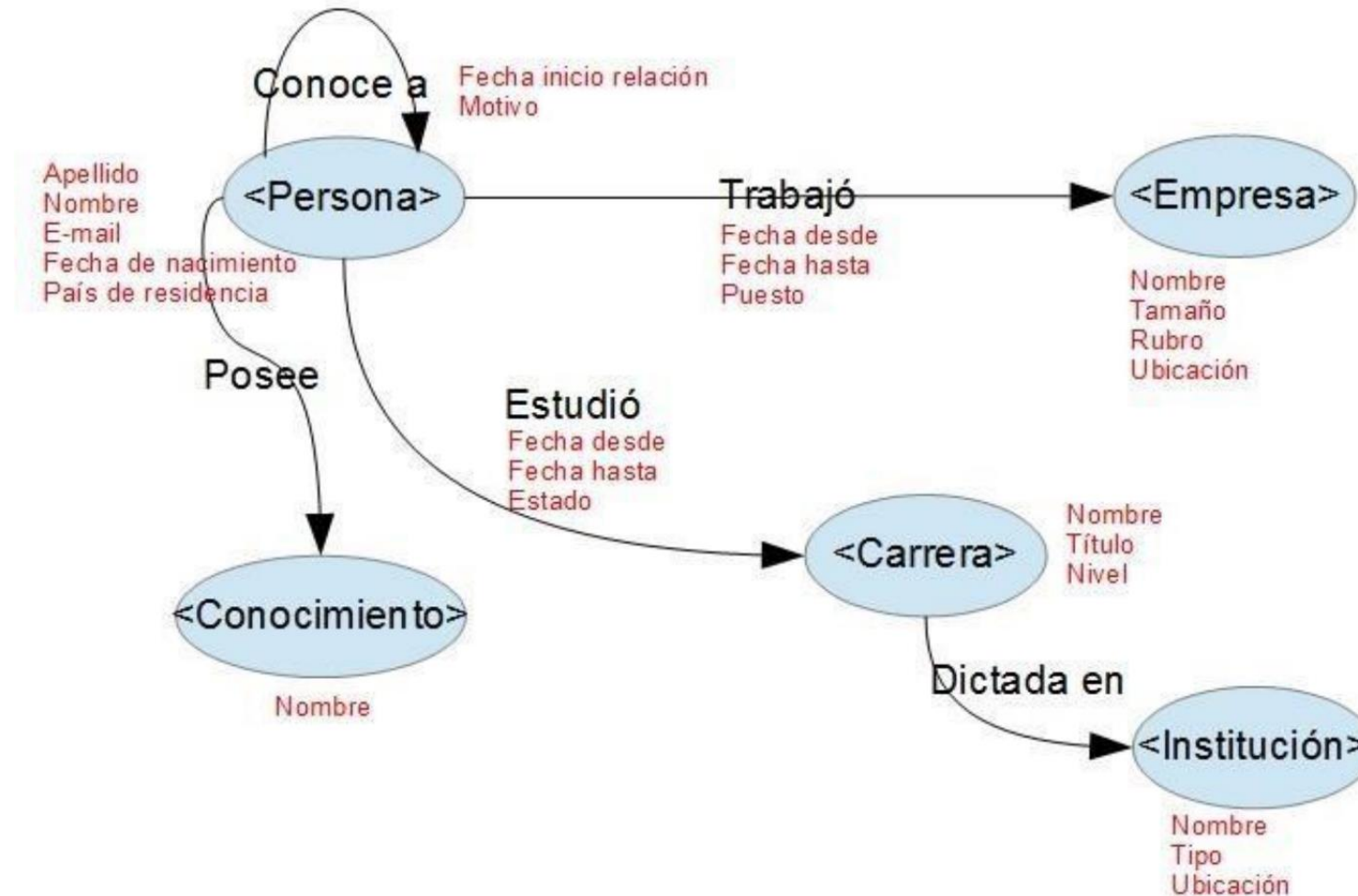
CASO PRÁCTICO: RED SOCIAL PROFESIONAL

Algunas consultas posibles

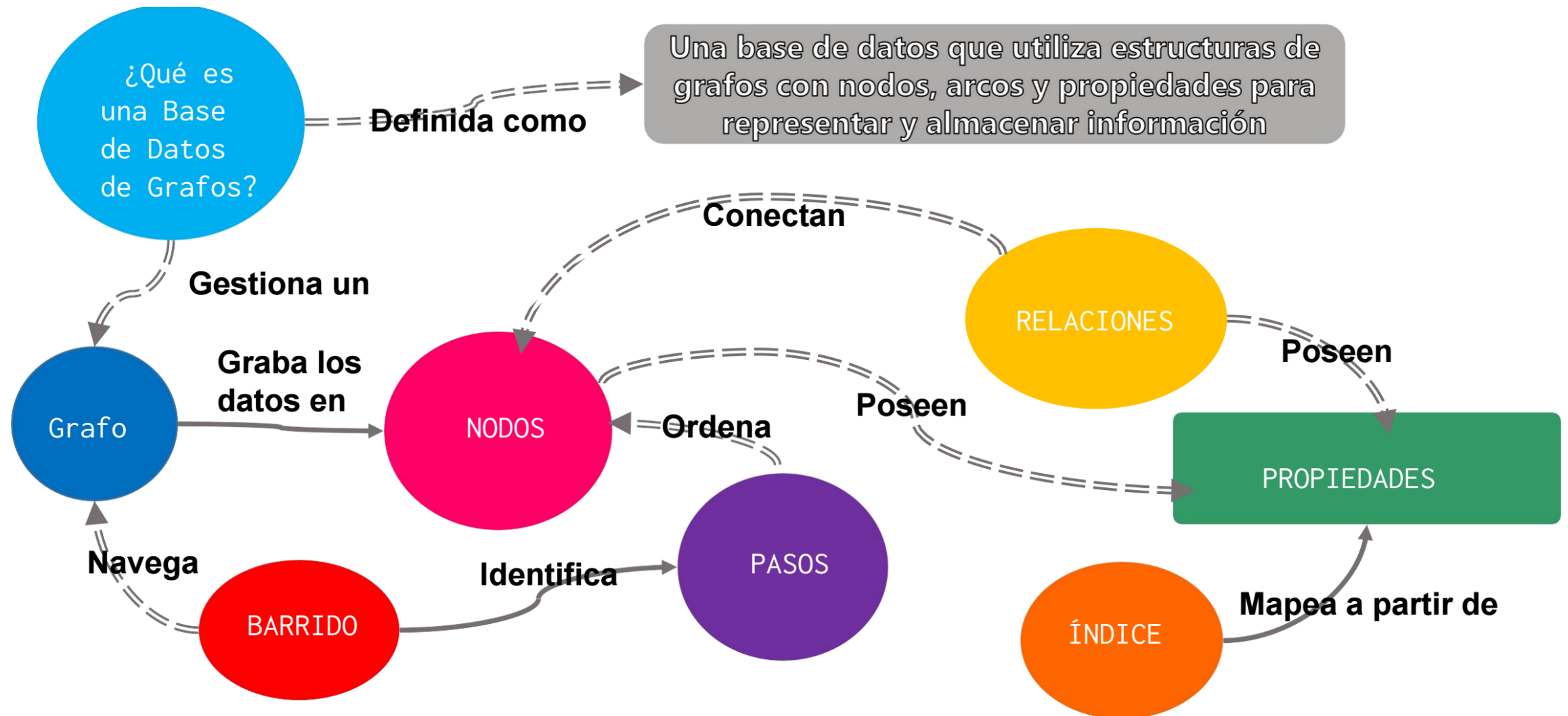
- Lista de personas que trabajan o trabajaron en empresas en las que una persona determinada trabajó, pero que no son sus contactos, para sugerirle nuevos contactos.
- Lista de personas que estudiaron o estudian en la misma institución que una persona determinada, pero que no son sus contactos, para sugerirle nuevos contactos.
- Ranking de los primeros 2 conocimientos que poseen más personas egresadas de la carrera "Ing en Sistemas de Información".
- Ranking de las 3 personas más populares: las que son conocidas por más personas en la red.

CASO PRÁCTICO: RED SOCIAL PROFESIONAL

Grafo de Propiedades



GRAPH BASED





GRAPH BASED

- Las bases de datos de grafos nos permiten almacenar entidades y relaciones sobre esas entidades.
- Las entidades las denominamos Nodos que tienen propiedades.
- Las relaciones denominadas Arcos pueden también tener propiedades.



GRAPH BASED

- **Las bases de datos de grafos nos permiten almacenar entidades y relaciones sobre esas entidades.**
- **Las entidades las denominamos Nodos que tienen propiedades.**
 - Podemos hacer una analogía entre un Nodo y un objeto instanciado en una aplicación
- **Las relaciones denominadas Arcos pueden también tener propiedades.**



GRAPH BASED

- Las bases de datos de grafos nos permiten almacenar entidades y relaciones sobre esas entidades.
- Las entidades las denominamos Nodos que tienen propiedades.
- Las relaciones denominadas Arcos pueden también tener propiedades.
 - Los arcos son direccionados



GRAPH BASED

- Los nodos están organizados por relaciones que nos permiten encontrar patrones entre estos.
- La organización del Grafo permite que los datos sean almacenados una vez y luego interpretados de distintas maneras basándonos en sus relaciones.

GRAPH BASED

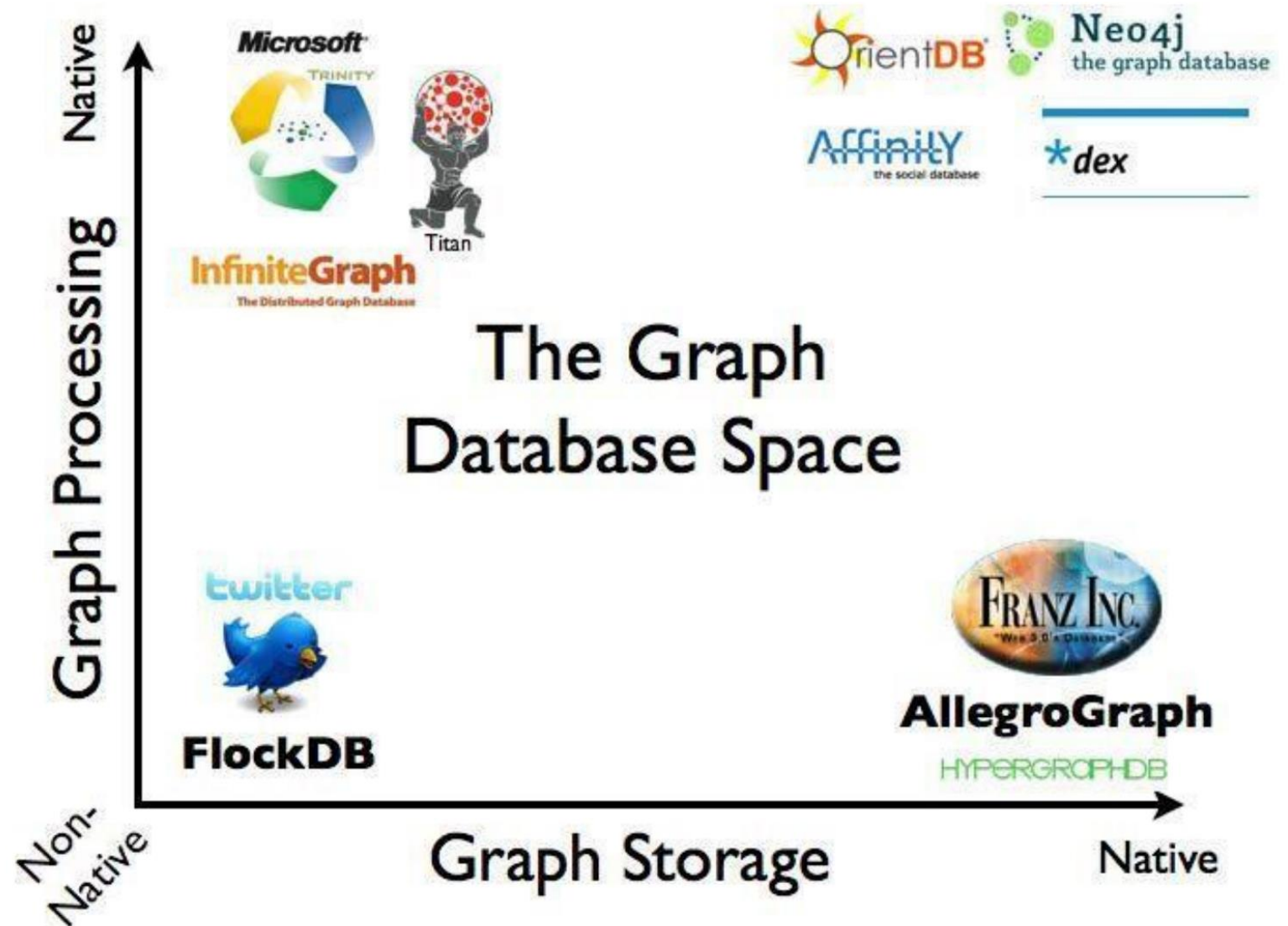
BASE DE DATOS	PUNTUACIÓN DE POPULARIDAD DB-ENGINES
NEO4J	52,06
ORIENTDB	4,76
ARANGODB	4,57
AMAZON NEPTUNE	2,90
DGRAPH	1,66
GIRAPH	1,59
INFINITE GRAPH	0,55
FLOCKDB	0,45

Fuente: www.db-engines.com

CATEGORÍAS DE DB BASADAS EN GRAFOS

Dos ejes:

- Almacenamiento interno del grafo
- Procesamiento del grafo



CONSULTA DE DATOS

- Las Bases de datos de Grafos soportan acceso a sus datos a través de lenguajes de consulta tales como Gremlin (lenguaje específico para recorrer grafos).
- Gremlin puede ser utilizado en todas las bases orientadas a Grafos que implementen “Blueprints property graph” (*).
- **Neo4J también cuenta con el lenguaje de consulta Cypher para recorrer grafos.**
 - *Cypher usa las keywords MATCH para búsqueda de patrones en relaciones, WHERE para filtrar propiedades en un nodo o relación, RETURN especifica qué retorna una consulta.*
 - *Cypher también provee métodos para operar los datos tales como ORDER, AGGREGATE, SKIP, LIMIT, entre otros.*

CONSULTA DE DATOS

- Aparte de estos lenguajes de consulta, Neo4J nos permite consultar las propiedades de los nodos, recorrer los grafos, o navegar por las relaciones de nodos utilizando “bindings” con diferentes lenguajes de programación (Java/.NET/JavaScript/Ruby/Python/PHP/R/Perl...).
- Índices. Se pueden crear índices sobre propiedades de un nodo.
- Se pueden aplicar filtros direccionales (entrantes, salientes ó ambos).

EN QUE CASOS USARLAS?

Datos interconectados:

- **Redes sociales** (likes, amigos, seguidores, etc.) ó laborales, por ej. representar a los empleados, sus conocimientos, y la relación de trabajo con otros empleados en diferentes Proyectos
- **Relaciones entre entidades de dominio de diferentes dominios** (por ejemplo, sociales, espaciales, comercio, redes y operaciones de IT, identidad y gestión de accesos) en una sola base de datos, puede hacer estas relaciones más valiosas, proporcionando la capacidad de recorrer a través de dominios.

Servicios de Ruteo, Despacho

- Cada ubicación en nuestra red de despacho es un nodo, las relaciones pueden contener la distancia entre las ubicaciones como propiedad.

EN QUE CASOS USARLAS?

- **Motores de Recomendaciones:**
 - Como los nodos y las relaciones se crean en el sistema, que pueden ser utilizados para hacer recomendaciones como "sus amigos también han comprado este producto" o "al facturar este artículo, estos otros artículos suelen ser facturados" en tiempo real.
- **Sistemas con búsquedas recursivas con n niveles.**
- **Búsqueda de patrones en las relaciones para detectar el fraude en las transacciones en tiempo real.**
- **Gestión de datos maestros: líneas de organización y producción que naturalmente se modelan como grafos.**

EN QUE CASOS NO USARLAS?

- **Sistemas que requieren de actualizaciones masivas sobre todas las entidades o un conjunto de entidades para un atributo o conjunto de atributos específicos.**
- **Sistemas que requieren una alta distribución de datos debido a su gran tamaño.**

ALGUNOS CASOS DE ÉXITO DE NEO4J

- Detección de fraude
- Búsqueda basada en Grafos
- Identidad y control de acceso
- Gestión de datos maestros
- Redes y Operaciones IT
- Recomendaciones en tiempo real
- Redes sociales

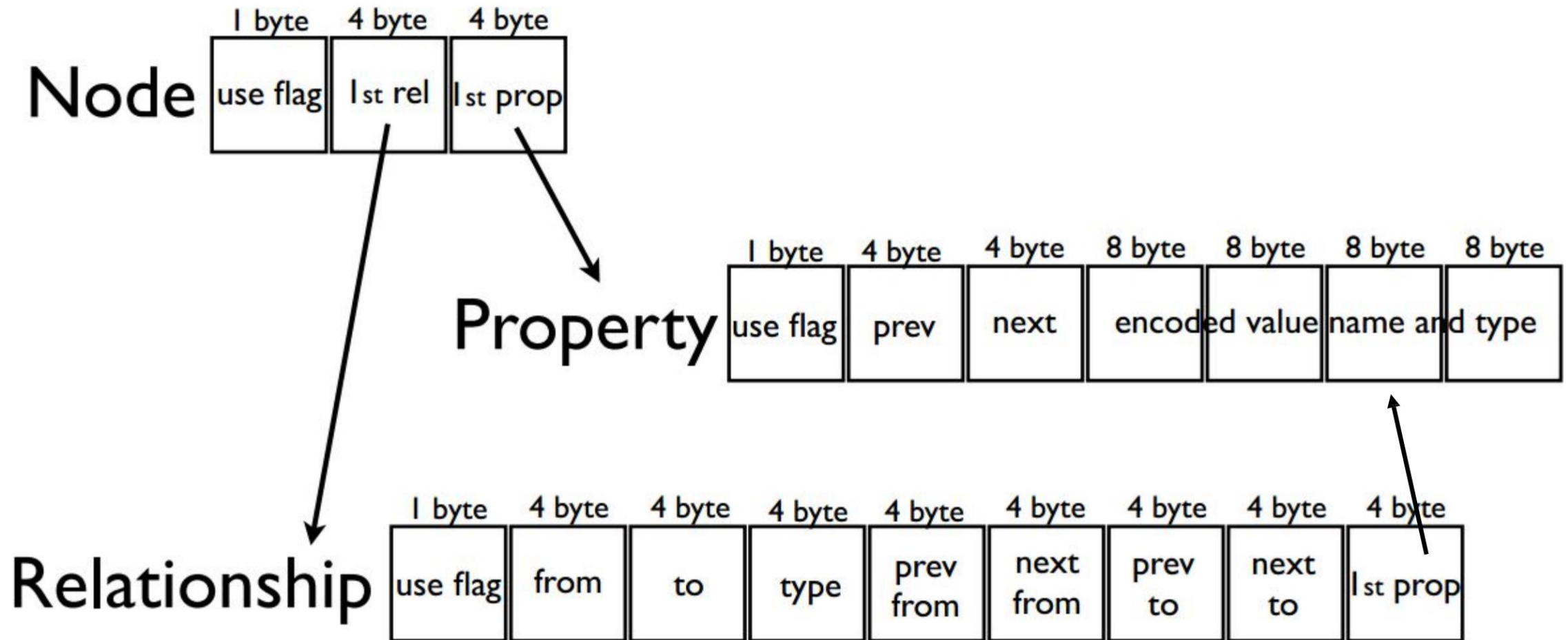


ESTRUCTURA INTERNA DE ALMACENAMIENTO

enUso
idPróximaRelación
idPróximaPropiedad
etiquetas
extra

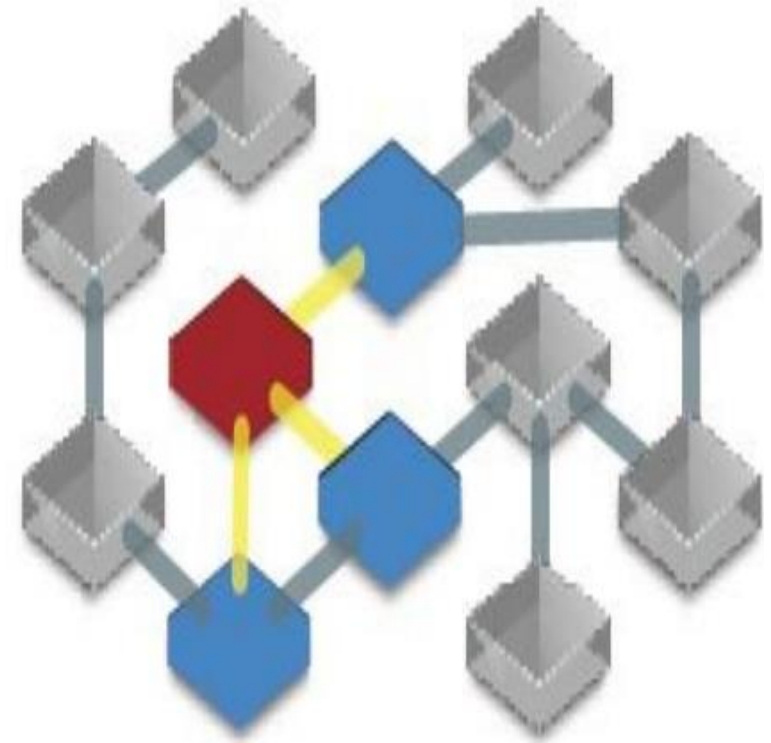
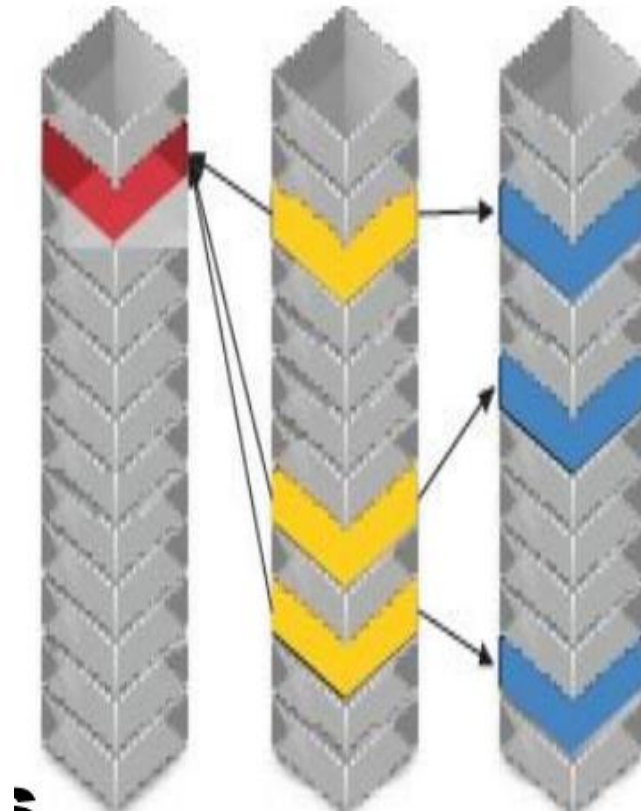
enUso	idPrimeraRelaciónPrevia	idPróximaPropiedad
primerNodo		
	idPrimeraRelaciónPróxima	primerMarcadorEnCadena
segundoNodo		
	idSegundaRelaciónPrevia	
tipoRelación		
	idSegundaRelaciónPróxima	

ESTRUCTURA INTERNA DE ALMACENAMIENTO



RDBMS VS GRAFOS

- **RDBMS:** referencias a otras filas y tablas referenciando sus atributos clave mediante columnas de clave foránea.
- **Joins:** uso intensivo de procesamiento y memoria y costo exponencial.
- Si se usan relaciones de muchos a muchos, se debe introducir una tabla de join que guarde las FK de ambas tablas involucradas, lo cual incrementa más aún el costo de las operaciones de join.



RDBMS VS GRAFOS

- A pesar de su nombre, las BDs relacionales no guardan las relaciones entre los datos. Por eso no son adecuadas para los datos altamente conectados de hoy en día.
- Señales de relaciones:
 - Gran número de Joins
 - Numerosos auto-joins (joins recursivos), comunes para representaciones de árbol o jerárquicas.
 - Frecuentes cambios de esquema
 - Queries lentos a pesar de haber sido tuneados
 - Resultados precomputados

Estos síntomas pueden indicar que se está intentando solucionar un problema de grafos con una BD relacional, donde el valor deriva de las relaciones entre los datos.

ARQUITECTURA MONGO

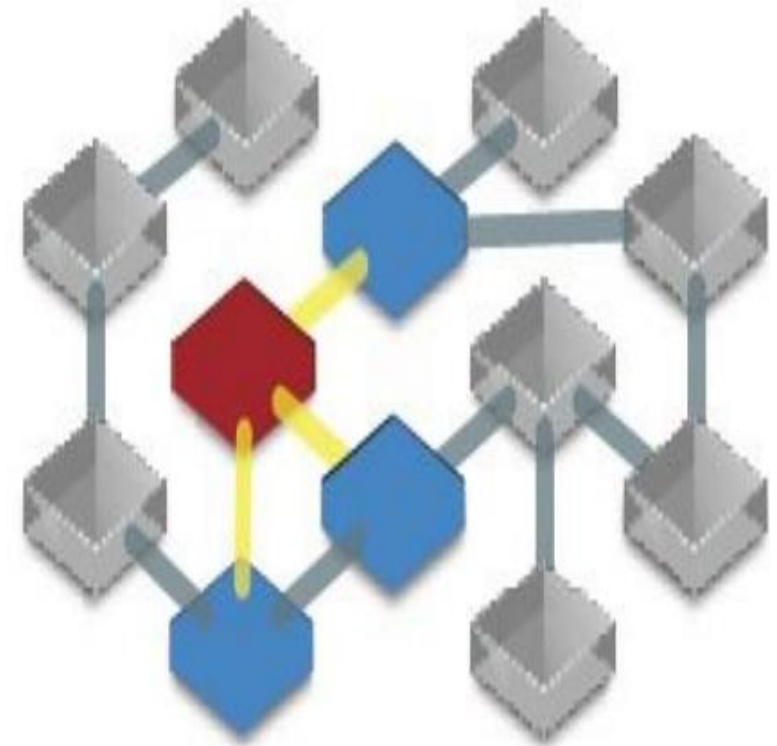
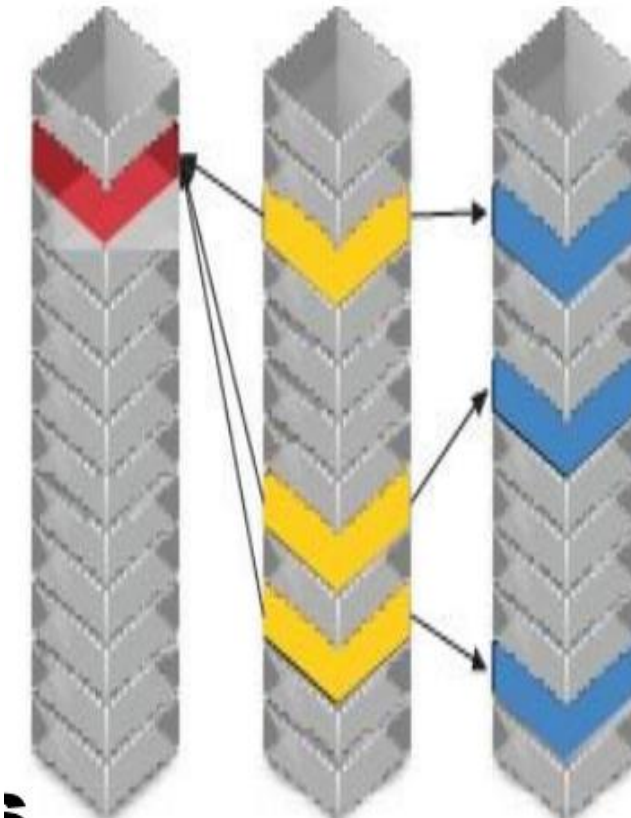
- Relaciones:

elementos privilegiados del modelo de grafos (sin necesidad de deducir las conexiones usando FK o procesamiento externo).

- El modelo de grafos permite construir modelos sofisticados que mapean exactamente al dominio del problema.

- Próxima generación de las RDBMS, pero con soporte de primera clase para las relaciones.

Equivalente a operación de join: recorrido de listas de relaciones con acceso directo a los nodos conectados.





FIN

PASEMOS A
LA PRÁCTICA