

Grafos con propiedades

Los nodos representan entidades (objetos)
Pueden pertenecer a una clase
Pueden tener propiedades que los describen
Los arcos representan relaciones entre entidades
Son dirigidos
Tienen una etiqueta
Pueden tener propiedades

Grafos con propiedades

El modelo físico puede variar
Incide en las consultas (pattern matching, path traversal)
Index free-adjacency models
No requiere índices para hacer eficiente la conectividad
Hace eficiente los recorridos sobre el grafo
Evita el alto costo de los joins
Permite recorrer las relaciones en ambas direcciones
Motores nativos de grafos

Neo4j

Base de datos orientada a grafos nativa
Almacenamiento de grafos
Procesamiento de consultas en grafos: recorrido de caminos
BD transaccional
ACID
Esquema flexible
Lenguaje de consulta: Cypher
Lenguajes: Java, C#, Python, Javascript, Ruby, PHP, R, Go

Cypher

Lenguaje de consulta declarativo
Describe que queremos seleccionar, insertar, actualizar, borrar
MATCH: el patrón a buscar
WHERE: agrega restricciones, filtra resultados intermedios
RETURN
CREATE, DELETE: nodos y arcos
SET, REMOVE: atributos
OPTIONAL MATCH

Cypher: Crear nodos

```
CREATE (variable:Etiqueta {attributeName1: value1, attributeName2: value2, ... });  
CREATE (p:Person { Name: 'Tim Burton' } );  
CREATE (m:Movie { Title: 'Planet of the Apes', Year:2001, Runtime: '119 min' } );
```

Cypher: Crear arcos

```
MATCH (n1: Etiqueta {atributo1:valor1}), (n2:Etiqueta {atributo2:valor})
CREATE (n1)-[r1:Relacion1]->(n2);
MATCH (p:Person { Name: 'Tim Burton' } ), (m:Movie {Title: 'Planet of the Apes' })
CREATE (m)-[r:Director]->(p)
```

Consultas: Path traversal

Navegar el grafo siguiendo los arcos que corresponden con un camino especificado
El recorrido inicia en uno o varios nodos seleccionados
En cada paso del recorrido se pueden tomar los valores de los atributos de los nodos o arcos o aplicar filtros sobre ellos

Cypher: Ejemplo

```
MATCH (m:Movie {Country: 'USA'})-[:Actor ]->(a), (m)-[:Director ]->(d), (m)-[:Rating ]->(ra)
RETURN m.Title, a.Name, d.Name, m.Year, ra.Source, r.Value;
```

Cypher: Optional match

```
MATCH (m:Movie {Country: 'USA'})-[:Actor]->(a), (m)-[:Director]->(d)
OPTIONAL MATCH (m)-[:Rating]->(ra)
RETURN m.Title, a.Name, d.Name, ra.Source, r.Value;
```

Cypher: Caminos de longitud variable

```
MATCH (n)-[:TYPE*minHoops..maxHoops]->(m) RETURN n, m;
MATCH (p:Person {Name: 'Harrison Ford'})-[:Actor*2]-(q) RETURN DISTINCT p.Name, q.Name;
MATCH (p:Person)-[:r:Actor*3..4]-(q) RETURN DISTINCT p.Name, q.Name;
```

Cypher: Otras cláusulas

```
MATCH (m:Movie), (p:Person) WHERE NOT (m)-->(p) RETURN m.Title, p.Name;
MATCH (p:Person {Name: 'Sam Robards'}),(q:Person {Name: 'Jesse Eisenberg'}), sp =
allShortestPaths((p)-[*]-(q)) RETURN sp;
```

Funciones propias de grafos

Grado del nodo
Número de arcos incidentes
Distancia entre dos nodos
Nodos alcanzables desde cierto nodo (reachability)
Camino mas corto entre dos nodos
Centralidad de un nodo
Importancia relativa de un nodo dentro del grafo
Diámetro del grafo

Neo4j: A tener en cuenta

No usar BLOBs
Guardar el URL que apunta al archivo
Balance entre ubicar todos los atributos y propiedades de un objeto en un solo nodo o separar cada

uno en nodos individuales
Puede afectar el desempeño de las consultas

Neo4j: Replicación

Replicación Maestro-Eslavo
Cluster manager: asegurar la BD y un cluster tolerante a fallos
Todo el grafo se replica en los esclavos
Escritura en el maestro
Se acepta por quorum
Lectura en cualquier nodo

Neo4j: Causal clustering

Dos tipos de nodos: Core servers y Replica
Read replica
Replica asíncrona
Causal consistency
Read-your-own-writes

Referencias

Robinson, Webber y Eifrem. Graph Databases. O'Reilly, 2013