Investigacion

Migrar una base transaccional a una de grafos especificamente de postgresql a neo4j

Lo primero que tendremos que hacer para obtener datos de una base de datos relacional en un gráfico es traducir el modelo de datos relacionales a un modelo de datos de gráfico. Determinar cómo queremos estructurar las tablas y filas como nodos y relaciones puede variar según lo que sea más importante para sus necesidades comerciales.

Al derivar un modelo de gráfico a partir de un modelo relacional, debemos tener en cuenta un par de pautas generales.

1. Una fila es un nodo
2. Un nombre de tabla es un nombre de etiqueta
3. Una combinación o clave externa es una relación

Con estos principios en mente, podemos mapear nuestro modelo relacional a un gráfico con los siguientes pasos:

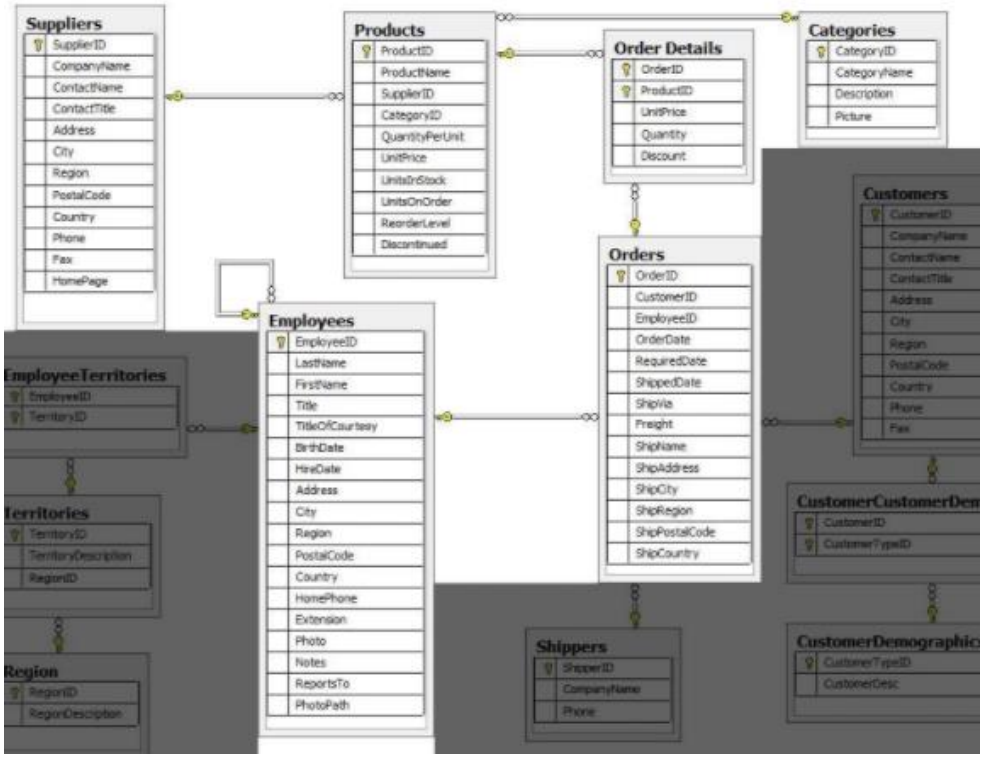


Fig1.Base de datos relacional

Filas a nodos, nombres de tablas a etiquetas

1. Cada fila de nuestra Orders tabla se convierte en un nodo en nuestro gráfico con Order la etiqueta.

2. Cada fila de nuestra Products tabla se convierte en un nodo con Product la etiqueta.

3. Cada fila de nuestra Suppliers tabla se convierte en un nodo con Supplier la etiqueta.

4. Cada fila de nuestra Categories tabla se convierte en un nodo con Category la etiqueta.

5. Cada fila de nuestra Employees tabla se convierte en un nodo con Employee la etiqueta.

Se une a las relaciones

1. Une entre Suppliers y se Products convierte en una relación nombrada SUPPLIES(donde el proveedor suministra el producto).

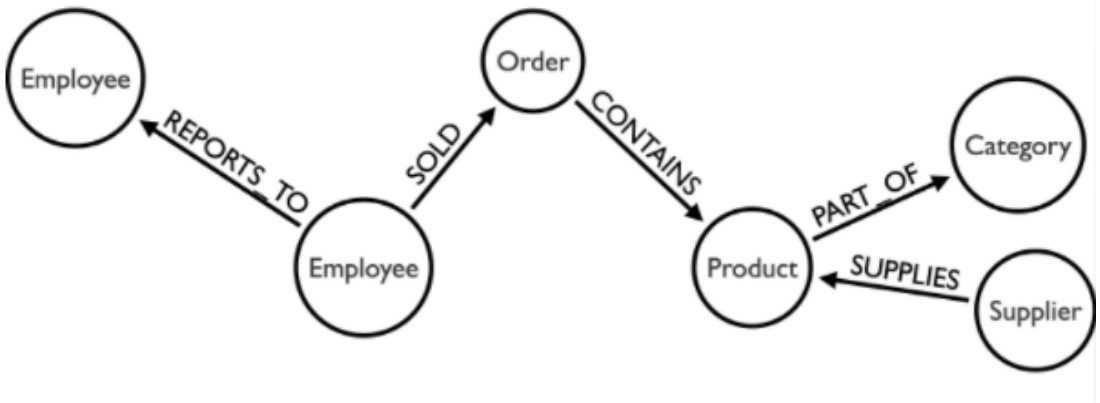
2. Unir entre Products y se Categories convierte en una relación nombrada PART_OF(donde el producto es parte de una categoría).

3. Unirse entre Employees y se Orders convierte en una relación nombrada SOLD(donde el empleado vendió un pedido).

4. La unión entre Employees sí (relación unaria) se convierte en una relación nombrada REPORTS_TO(donde los empleados tienen un gerente).

5. Unir con tabla de unión (Order Details) entre Orders y Products se convierte en una relación llamada CONTAINS con propiedades de unitPrice, quantity y discount(donde pedido contiene un producto).

Si implementamos en Neo4j se vería de la siguiente manera:



Exportación de tablas relacionales a CSV

Un formato común que muchos sistemas pueden manejar en un archivo plano de valores separados por comas (CSV), así que veamos cómo exportar tablas relacionales desde una base de datos PostgreSQL a archivos CSV para que podamos crear nuestro gráfico.

El comando 'copiar' de PostgreSQL nos permite ejecutar una consulta SQL y escribir el resultado en un archivo CSV. Podemos ensamblar un breve script .sql de estos comandos de copia, como se muestra a continuación.

export_csv.sql sql

```
COPY (SELECT * FROM customers) TO '/tmp/customers.csv' WITH CSV header;

COPY (SELECT * FROM suppliers) TO '/tmp/suppliers.csv' WITH CSV header;

COPY (SELECT * FROM products)  TO '/tmp/products.csv' WITH CSV header;

COPY (SELECT * FROM employees) TO '/tmp/employees.csv' WITH CSV header;

COPY (SELECT * FROM categories) TO '/tmp/categories.csv' WITH CSV header;

COPY (SELECT * FROM orders

   LEFT OUTER JOIN order_details ON order_details.OrderID = orders.OrderID) TO '/tmp/orders.csv' WITH CSV header;
```

Luego, podemos ejecutar ese script en nuestra base de datos Northwind con el comando psql -d northwind < export_csv.sql, y creará los archivos CSV individuales enumerados en nuestro script.

Importando los datos usando Cypher

Después de haber exportado nuestros datos desde PostgreSQL, usaremos el comando LOAD CSV de Cypher para transformar el contenido del archivo CSV en una estructura de gráfico. Primero, probablemente querremos colocar nuestros archivos CSV en un directorio de fácil acceso. Con Neo4j Desktop, podemos colocarlos en el directorio de importación de la base de datos local (instrucciones detalladas que se encuentran en nuestra guía de importación de escritorio). De esta manera, podemos usar el file:///prefijo en nuestras declaraciones Cypher para ubicar los archivos. También podemos colocar los archivos en otro directorio local o remoto (admite HTTPS, HTTP y FTP) y especificar la ruta completa en nuestras declaraciones Cypher. Dado que estamos usando Neo4j Desktop en este ejemplo, usaremos la carpeta de importación para la base de datos y la ruta de nuestros archivos CSV puede comenzar con el file:///prefijo.

Ahora que tenemos nuestros archivos a los que podemos acceder fácilmente, podemos usar el LOAD CSVcomando de Cypher para leer cada archivo y agregar declaraciones Cypher después para tomar los datos de fila / columna y transformarlos en el gráfico.

```
// Create orders
LOAD CSV WITH HEADERS FROM 'file:///orders.csv' AS row
MERGE (order:Order {orderID: row.OrderID})
```

```
  ON CREATE SET order.shipName = row.ShipName;

// Create products
LOAD CSV WITH HEADERS FROM 'file:///products.csv' AS row
MERGE (product:Product {productID: row.ProductID})
  ON CREATE SET product.productName = row.ProductName,
product.unitPrice = toFloat(row.UnitPrice);
```