TIBCO®

How to Map Relational Data to a Graph DB in Four Steps

By Steven Yang

Today, the relational data storage model is probably the most popular concept for storing data. Just like its physical ancestor, index cards, relational data storage groups data records of the same type (those with the same properties or attributes). Storing data this way makes single record information retrieval easy because all attributes for a record are stored in the same location. Searching among large set of records can be very effective using an attribute indexing mechanism.

Considering database history, it's not surprising that most of today's data is still stored in relational models. However, in the relational data storage model, there is no physical link to indicate the relation among different data records. They are only linked logically by a special attribute called a foreign key, rendering the search for related records requires lookups on the entire target table, which as the target table grows, is very inefficient.

THE GRAPH DATABASE SOLUTION

On the other hand, the graph data storage model keeps data attributes together in nodes. By creating physical links called edges, all related nodes are kept together as neighbors. As long as the number of related nodes remains the same, the performance of retrieving them stays the same, even though the number of nodes in other partitions of the graph increase dramatically. Finding related nodes in a graph database is much more efficient than finding related data rows in a relational database model.

A new class of employees—mostly business analysts, data scientists, and marketing strategists—are now trying to gain better insight into their data using the relationships between their enterprise data sets. To better serve these new ways of searching for information, it is becoming critical to find a way to expose existing enterprise data in a graph database format. In this paper, we'll use a well-known example to demonstrate an approach to map information from relational data to a graph database.

HOW IT WORKS

Mapping data from a relational database to graph database is fundamentally a task of converting the relational representation from one database to the other, from table structure to graph structure. A data entity in the table of the relational database is a "row (that contains columns), and in the graph database it's a node (that contains attributes). More specifically, we can use the foreign keys of the relational data model to build edges, thus transforming loosely coupled data records into a highly bounded group of nodes.

Let's start with a simplified knowledge domain created from a Microsoft Northwind database sample and go step by step on how to map the data from a relational to a graph database.

Figure 1 shows five tables that are a modified version of the Microsoft Northwind database, a typical relational data model.

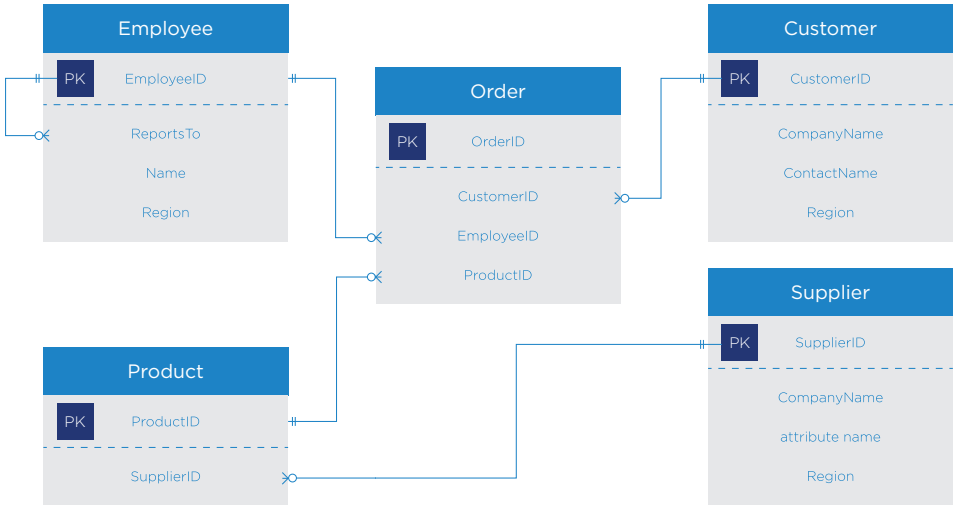

Figure 1. Relational data schema diagram for Simplified Northwind db

STEP ONE

The first step is to identify the data entity types. In this example there are five types of data entities:

- Employee
- Order
- Customer
- Product
- Supplier

Each table represents the different types of information we are manipulating.

STEP TWO

In the second step, we need to find the semantic relations between those entities. We find five relations among them which are:

RELATION	LOGICAL LINK (ATTRIBUTE)
Order - SoldBy -> Employee	EmployID
Order - SoldTo -> Customer	CustomerID
Order -> SoldItem -> Product	ProductID
Supplier - Supplies -> Product	SupplierID
Employee - ReportsTo -> Employee	EmployeeID

This step is less obvious. As we mentioned before, the relational database is not handling relations as a first class citizen. “Reports To” relation can be found as an attribute in Employee table, but “Sold By” semantic is not in the schema. There is a logical link between Order and Employee using a foreign key, but the meaning of this relation is not coded in the database.

See figure 2. The relations on the schema diagram are highlighted.

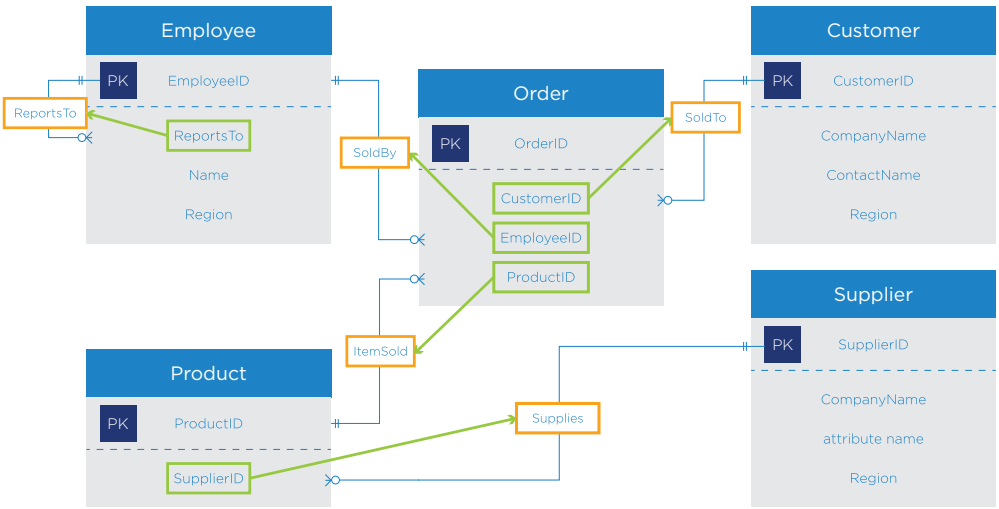

Figure 2. Identify relation between data

STEP THREE

When data entities and relations (in above table and Figure 2) are correctly identified, we can create nodes for each data entity and edges for each relation we just discovered. Now we’ve created a graph from the relational data model based on the predefined relation (foreign key in relational data schema).

Figure 3. In a graph, all related nodes (data records in relational data model) are physically linked by the edges.

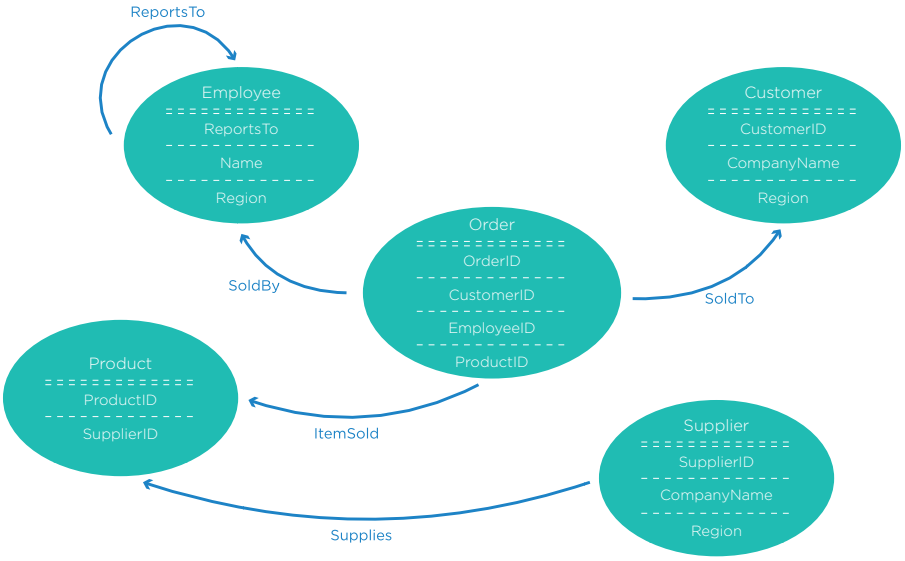

Figure 3. Convert Relational Data to Graph Node/Edge

Let's continue a step further.

STEP FOUR

In Figure 4, you can see that there are two common attributes that are used in more than one node. Region information is used in Employee, Customer, and Supplier. And Company information is used in Customer and Supplier.

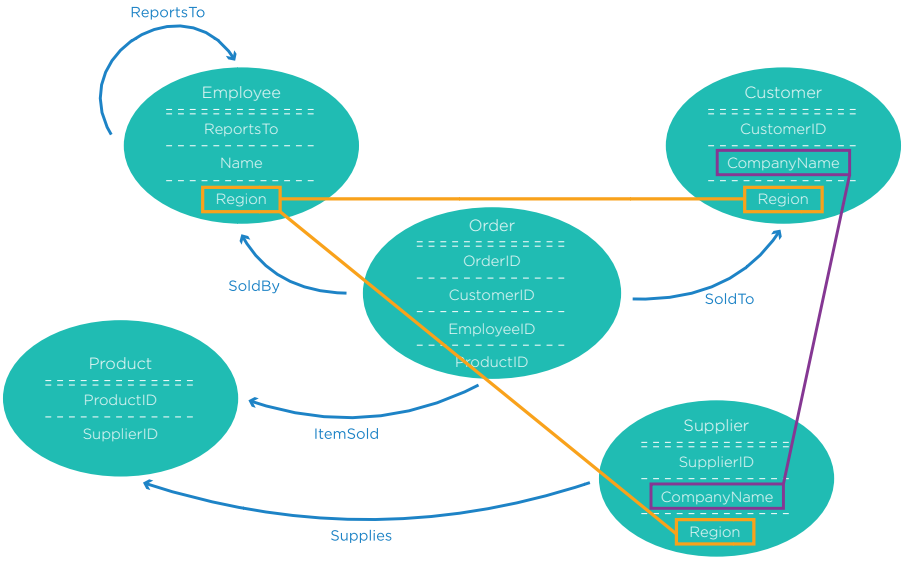

Figure 4. Find same type of attribute in different node

Promoting those common attributes to nodes with proper relations to the entities could add value to our graph. In our example, after promoting Region and Company to nodes, we can easily answer questions like the ones below:

- Question 1: Give me all customers that are in the same region as James Bond.
- Question 2: Which companies are both our customer and our supplier?

Figure 5. Shows that the graph has two new nodes Region and Company.

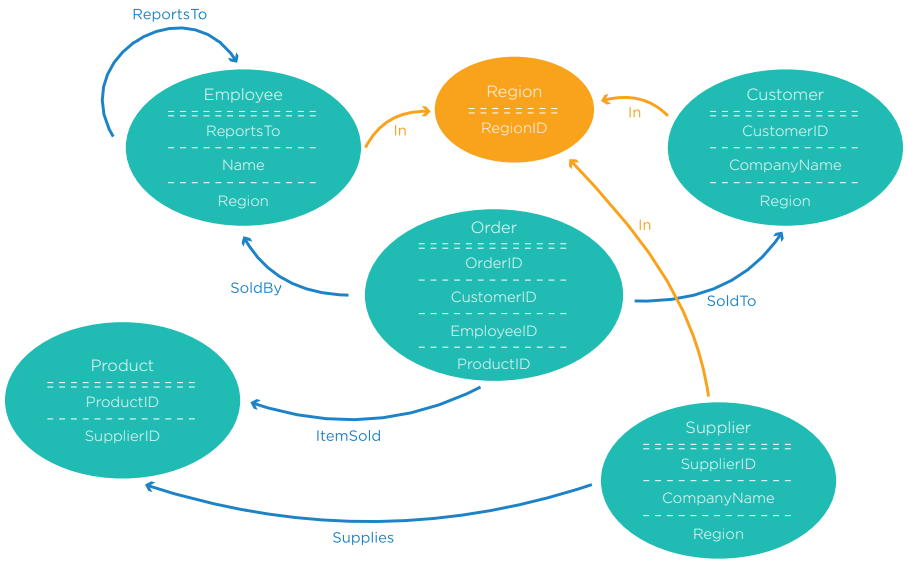

Figure 5. Promote Common Attributes to Nodes

Finding and building the right edges in the graph is a key activity. These steps impact the ability of the data structure to respond efficiently to end users' questions. They consist of transforming logical relations (foreign keys) and implicit relations (common attributes) into explicit edges that are physically stored in graph structure. We explained this approach using a very simple use case. With real enterprise data, we would have to use specific tools to facilitate the design of this mapping and to automate the data migration. We will describe what such a tool could look like in a following paper.

TIBCO®

Global Headquarters
3307 Hillview Avenue
Palo Alto, CA 94304
+1 650-846-1000 TEL
+1 800-420-8450
+1 650-846-1005 FAX
www.tibco.com

03/15/18