Powering Real-Time Recommendations

with Graph Database Technology

Jim Webber, Chief Scientist, Neo4j

White Paper

TABLE OF CONTENTS

"We found Neo4j to be literally thousands of times faster than our prior MySQL solution, with queries that require 10-100 times less code."

– Volker Pacher, Senior Developer, eBay

Powering Real-Time Recommendations

with Graph Database Technology

Jim Webber, Chief Scientist, Neo4j

"You May Also Like..."

"You may also like" is a deceptively simple phrase that encapsulates a new era in customer relationship management. In offering tailored suggestions, businesses maximize the value they deliver by providing highly targeted, real-time product recommendations to their online consumers.

This ability to make compelling offers requires a new generation of technology. That technology must capture the customer's buying history and also instantly analyze their current choices, before immediately matching them to the most appropriate product recommendations. And all of this analysis must be done in real time before the customer moves to a competitor's website.

The key technology in enabling real-time recommendations is the graph database, a technology that is quickly leaving traditional relational databases (RDBMS) behind.

Graph technology easily outperforms relational and other NoSQL data stores when it comes to connecting masses of buyer and product data (and connected data in general) to gain insight into customer needs and product trends.

Significantly, graph databases are a core technology platform of internet giants like Google, Facebook and LinkedIn. But while those pioneers had to build their own in-house data stores from scratch, off-the-shelf graph databases – notably Neo4j – are now available to any business wanting to make the most of real-time recommendation engines.

The profit and productivity improvements graph technology offers over relational databases are astounding. eBay uses the Neo4j Graph Platform for a number of different projects, including a real-time recommendation engine.

"We found Neo4j to be literally thousands of times faster than our prior MySQL solution, with queries that require 10-100 times less code," said Senior Developer Volker Pacher. "Today, Neo4j provides eBay with functionality that was previously impossible."

Graph Technology: An Uncontroversial Choice

eBay is not alone in selecting an off-the-shelf graph database like Neo4j as a core platform for business-critical systems. Neo4j is the world's most popular graph database, according to database monitoring site DB-Engines. Graph databases are growing in popularity faster than any other type of database – they nearly multiplied their popularity by six times in a three-year period. The DB-Engines authors have noted in the past that "graph databases are grabbing an ever-larger slice of developers' attention. If you haven't used them yet, perhaps it's time to have a closer look."

The key to understanding graph database systems is that they give equal prominence to storing both the data (customers, products, categories) and the *relationships* between them (who bought what, who likes whom, which purchase happened first). In a graph database, you don't have to live with the semantically poor data model and expensive, unpredictable JOINs from the relational database world.

Instead, graph technology supports many named, directed relationships between entities (or nodes) which give a rich semantic context for the data. Now we can specify both **`loves`** and **`married to`** a spouse, **`owns`** and **`dislikes`** a game console, or repeatedly **`visited`** a store that was amazing. And queries are super-fast since there is no JOIN penalty.

This makes graph databases especially suited to formulating real-time recommendations, because making the best recommendations – and maximizing value – involves more than simply offering products because they are best sellers. Best sellers can be a successful part of a recommendation, but they are one which by their nature are an aggregate picture of all customers. Nowadays customers expect finely-tuned recommendations in the long tail and they react poorly to one-size-fits-all suggestions.

Real-time recommender systems require the ability to understand the customer's past purchases, quickly query this data and match the customer to the people who are the closest match to them both in their social network and in buying patterns. To make real-time recommendations also requires the ability to instantly capture any new interests shown in the customer's current visit. Matching historical and session data like this is trivial for Neo4j.

The Business Benefits of Neo4j

Companies from around the globe have incorporated Neo4j into their data architecture to take advantage of the powerful real-time recommendations the Neo4j Graph Platform provides. These enterprises have experienced a variety of business benefits as a direct result.

Improved Competitiveness

Neo4j enables new types of business functionality that are often not possible with other technologies, allowing you to make real-time decisions based on connected data. For example, Walmart uses Neo4j to make real-time product recommendations by using information about what users prefer. Additionally, many dating and online job sites use Neo4j to recommend jobs or dates by incorporating knowledge of the extended network (friends-of-friends) into the recommendation, again in real time, substantially improving the accuracy of the recommendation.

Reduced Project Time and Cost

Neo4j cuts the overhead on many types of projects, particularly those involving connected data. Many customers cite the huge acceleration that occurs when a graph model is brought to bear on a connected data problem. For example, eBay cites that with Neo4j it requires 10-100 times less code than it did with SQL, and Telenor, one of the world's top telecom providers, uses Neo4j for the authorization system on its business customer portal, improving performance by 1,000 times.

> "Neo4j helps us to understand our online shoppers' behavior and the relationships between our customers and products, providing a perfect tool for real-time product recommendations. As the current market leader in graph databases, and with enterprise features for scalability and availability, Neo4j is the right choice to meet our demands."
>
> *– Marcos Wada, Software Developer, Walmart*

"Neo4j allowed us to add functionality that was previously not possible."

– Volker Pacher, Senior Developer, eBay

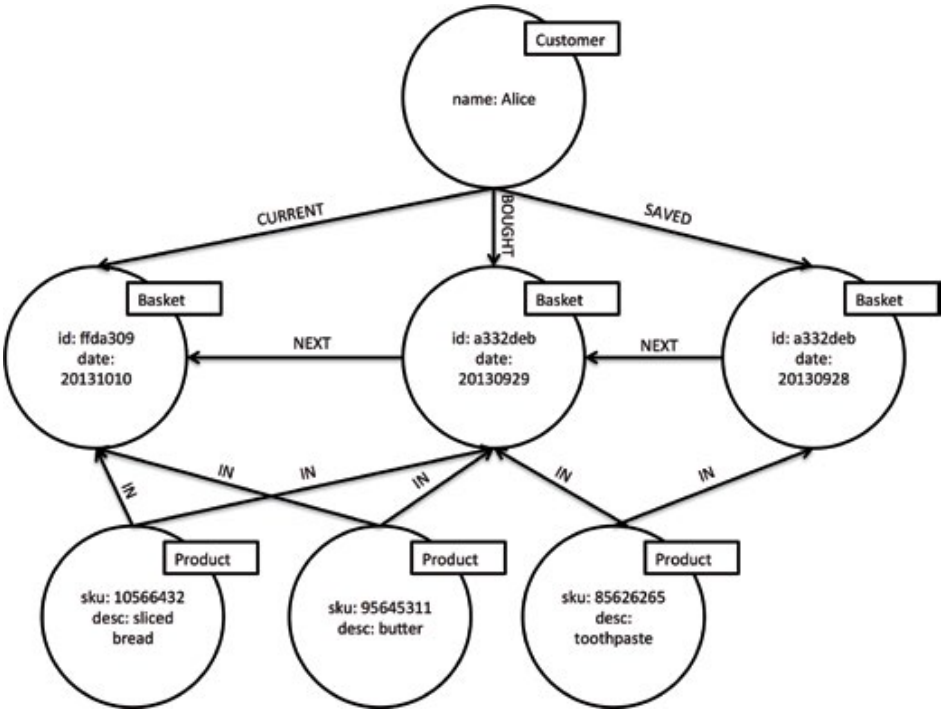
Faster Product Time to Market and Better Performance

Neo4j requires developers to produce less code than RDBMS alternatives. Less code means higher quality and an increased success rate on projects. Neo4j's performance is dramatically better for connected datasets – often the difference between what will and won't be possible for a development team.

The eBay team states that "Neo4j allowed us to add functionality that was previously not possible." Many customers experience improvements on a similar scale, so much so that Neo4j is often described as decreasing query times from "minutes to milliseconds" for connected data queries.

Rapid Example: A Retail Recommendation Engine using Neo4j

In a retail scenario (either online or brick-and-mortar), we could store the baskets that customers have purchased in a graph like the one below.



This graph shows how we use a simple linked list of shopping baskets connected by **NEXT** relationships to create a purchase history for the customer.

In that graph, we see that the customer has visited three times, saved their first purchase for later (the **SAVED** relationship between customer and basket nodes) and ultimately bought one basket (indicated by the **BOUGHT** relationship between customer and basket node) and is currently assembling a basket, shown by the **CURRENT** relationship that points to an active basket at the head of the linked list.

It's important to understand this isn't a schema or ER-diagram but represents actual data for a single customer. A real graph of many such customers would be huge (far too big for examples in a white paper) but would exhibit the same kind of structure.

In graph form, it's easy to figure out the customer's behaviour: they became a (potential) new customer but failed to commit to buying toothpaste and came back one day later and bought toothpaste, bread and butter. Finally, the customer settled on buying bread and butter in their next purchase – which is a repeated pattern in their purchase history we could ultimately use to serve them better.

Now that we have a graph of customers, and the past products they've bought, we think about recommendations to influence their future buying behaviour. By far the simplest recommendation is to show popular products across the store. This is trivial in Cypher as we see in the following query:

```
MATCH (customer:Customer)-[:BOUGHT]->(:Basket)
      <-[:IN]-(product:Product)
RETURN product, count(product)
ORDER BY count(product) DESC LIMIT 5
```

The Cypher query above showcases much about Cypher.

First, the **MATCH** clause shows how ASCII-art is used to declare the graph structure (or pattern) that we're looking for. In this case, it can be read as "customers who bought a basket that had a product in it" except since baskets aren't particularly important for this query we've elided them using the anonymous node **()**.

Then we **RETURN** the data that matched the pattern and operate on it with some (familiar-looking) aggregate functions. That is, we return the node representing the product(s) and the count of how many product nodes matched, then order by the number of nodes that matched in a descending fashion, limiting to the top five which gives us the most popular products in the purchasing data

However, this query isn't really contextualized by the customer but by *all* customers and so isn't optimized for any given individual (though it might be very useful for supply chain management). We do better without much additional work by recommending historically popular purchases that the customer has made themselves, as in the following query:

```
MATCH (customer:Customer {name: 'Alice'})-[:BOUGHT]
       ->(:Basket)<-[:IN]-(product:Product)
RETURN product, count(product)
ORDER BY count(product) DESC LIMIT 5
```

The only change in this query, compared to the previous, is the inclusion of a constraint on the customer node that it must contain a key name and a value **Alice**. This is actually a far better query from the customer's point of view since it's ego-centric (as good recommendations should be!).

"We've seen a 300% growth in the price changes that properties are generating. This has driven a significant amount of business growth."

– Senior Director,
Fortune 200 Hospitality Chain

Of course in an age of social selling, it'd be even better to show the customer popular products in their social network rather than just their own purchases since this strongly influences buying behaviour. As you'd expect, adding a social dimension to a Neo4j graph is easy, and querying for friends/friends-of-friends/neighbors/colleagues or other demographics is straightforward as in this query:

```
MATCH (customer:Customer {name: 'Alice'})-[:FRIEND*1..2]->(friend:Customer)
WHERE customer <> friend
WITH DISTINCT friend
MATCH (friend)-[:BOUGHT]->(:Basket)<-[:IN]-(product:Product)
RETURN product, count(product)
ORDER BY count(product) DESC LIMIT 5
```

To retrieve the purchased products of both direct friends and friends-of-friends, we use the Cypher **`WITH`** clause to divide the query into two logical parts, piping results from the first part into the second. In the first part of this query, we see the family syntax where we find the current customer (**`Alice`**) and traverse the graph matching for either Alice's direct friends or their friends (her friend-of-friends).

This is a straightforward query since Neo4j supports a flexible path-length notation, like so: **`-[:FRIEND*1..2]->`** which means one *or* two **`FRIEND`** relationships deep. In this case we get all friends (depth one) and friend-of-friend (at depth two), but the notation can be parameterized for any maximum and minimum depth.

In matching, we must take care not to include Alice herself in the results (because your friend's friend is you!). It is the **`WHERE`** clause which ensures there is only a match when the customer and candidate friend are not the same node.

We don't want to get duplicate friends-of-friends that are also direct friends (which often happens in groups of friends). Using the **`DISTINCT`** keyword ensures that we don't get duplicate results from equivalent pattern matches.

Once we have the friends and friends-of-friends of the customer, the **`WITH`** clause pipes the results from the first part of the query into the second. In the second half of the query, we're back in familiar territory, matching against customers (the friends and friends-of-friends) who bought products and ranking them by sales (the number of bought baskets each product appeared in).

Graph technology even enables you to incorporate customer feedback, adjust for seasonal trends or suggest birthday gift ideas based on data on the customer's Facebook friends. And all in real time, without clever coding, and with no fear of the relational JOIN bomb.

Walmart and Other Leading Adopters

The following businesses are market leaders who are using Neo4j to serve up real-time recommendations in areas such as retail, industrial spare parts, jobs, movies, entertainment, restaurants and even online dating.

- Walmart calls Neo4j "a perfect tool for real-time product recommendations." The retailer has sales of more than $482 billion and employs 2.3 million associates worldwide, serving more than 260 million customers weekly through its 11,500 stores in 28 countries and ecommerce websites in 11 countries. Walmart Software Developer Marcos Wada explained: "Neo4j helps us to understand our online shoppers' behavior and the relationship between our customers and products, providing a perfect tool for real-time product recommendations. As the current market leader in graph databases, and with enterprise features for scalability and availability, Neo4j is the right choice to meet our demands."
- A leading movie recommendation website is revolutionizing the way the film industry promotes projects by enabling fans to discover the best upcoming releases before they hit the big screen and make recommendations based on individual taste. In turn, it provides movie studios with insights into the preferences and behavior of film fans, enabling them to more effectively target their marketing campaigns. They considered MySQL for its recommendation system, but after seeing the amount of data required, they looked at other databases and chose Neo4j. Their CTO said: "We wanted to quickly connect audiences to the right movies, and Neo4j just fits our philosophical standpoint. We are happy that we discovered Neo4j. We increased the speed of generating recommendations and users to movies, which is a core part of our business model."

Whether it's providing same-day delivery, real-time online product recommendations, or offering a powerful price-comparison tool, Neo4j and the real-time recommendations it affords is a driving force of success.

- eBay uses the delivery coordination platform Shutl to make the delivery of online and mobile orders quick and convenient. This eliminates the biggest roadblock between retailers and online shoppers: the option to have your item delivered the same day. Switching from MySQL to Neo4j allowed the fast-growing service to quickly match orders with couriers with relatively constant performance, and in a data model that allows queries to remain localized to their respective portions of the graph. "We achieved constant query performance by using Neo4j to create a graph that is its own index. That's awesome development flexibility," said Pacher.
- Wobi is a price comparison website for pensions and insurance that uses detailed financial pictures of customers to provide the best "value offers" to users. To achieve such a detailed level of customer understanding, Wobi needed a single customer database where it could rapidly drill down into each individual's history and add new information on the fly - which is exactly the model that Neo4j provides. Neo4j is currently handling half a million customers with an average of eight pensions and insurance policies and products each – a total of 4 million nodes and 30 million relationships. It has the capacity to expand much further. According to Shai Bentin, Chief Technology Officer at Wobi, "It's not a large database yet – but it will be. And I feel safe with Neo4j."
- An international Fortune 200 hospitality company adopted Neo4j to power its real-time pricing recommendation engine after running into significant slowdowns with its prior database architecture. Since working with Neo4j, the company has significantly reduced request processing time as well as hardware requirements, and performance has improved so substantially that they have seen a 300% growth in the volume of generated price changes.

Trying Out Neo4j

Neo4j is used by thousands of companies around the world, including more than 50 of the Global 2000 such as eBay, Walmart, Hewlett-Packard and Cisco. These companies have all recognized the value in – and necessity of – finding and leveraging connections between data for a variety of uses that ultimately provide customers with better user experiences.

Whether it's eBay providing customers with more seamless same-day delivery or accurate real-time product recommendations, or a well-known Fortune 200 hotelier providing a more powerful price-comparison tool, Neo4j and the real-time recommendations it provides are a consistent driving force of success.

Ready to see what Neo4j can do for your company? Learn how to master the emerging world of graph databases by reading the free O'Reilly book, *Graph Databases*, have your development team take Neo4j for a spin, and explore the variety of available online training options to get up and running with Neo4j in no time.

Neo4j is the leader in graph database technology. As the world's most widely deployed graph database, we help global brands – including Comcast, NASA, UBS, and Volvo Cars – to reveal and predict how people, processes and systems are interrelated.

Using this relationships-first approach, applications built with Neo4j tackle connected data challenges such as analytics and artificial intelligence, fraud detection, real-time recommendations, and knowledge graphs. Find out more at neo4j.com.

Questions about Neo4j?

Contact us around the globe:
info@neo4j.com
neo4j.com/contact-us