1. **Content based Recommendations:**

Pull everything from Product Catalog database into Neo4j and model it as a graph.

Create a schema for the graph database, like: Designer <-Product -> Category

Lets say a user on web application is searching for some product. We use that information to filter out the same category product from the above formed graph.

**Process:** Take the Id of product the user is searching -> Find its category -> find products of similar category from the graph database created by pulling the product catalog. To make more complex, we can show the products of the same category and those provided by the same seller. (eg: Category may be shoes and then seller may be adidas)

Following mappings can be done:

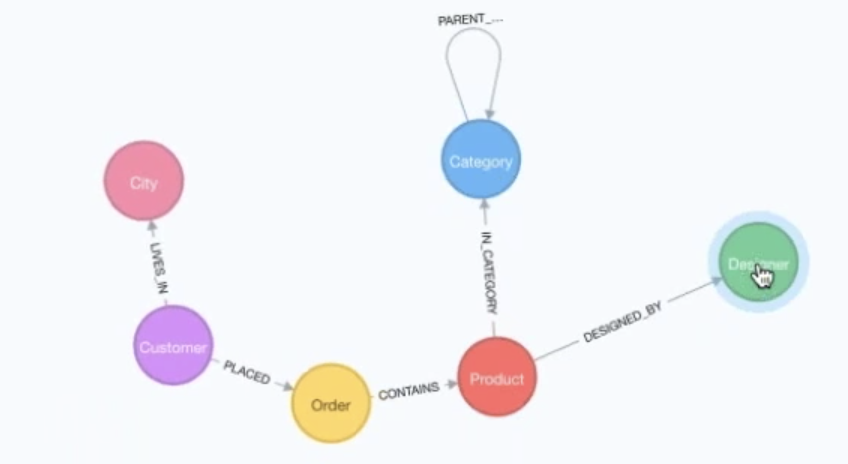
1. Recommend from same category
2. Same category and overlapping seller or designer
3. Walking hierarchy of categories (if exists)
4. If a user a purchased the desk, don’t recommend desk again, but something that falls into similar chain of category.

**Schema till now: .**

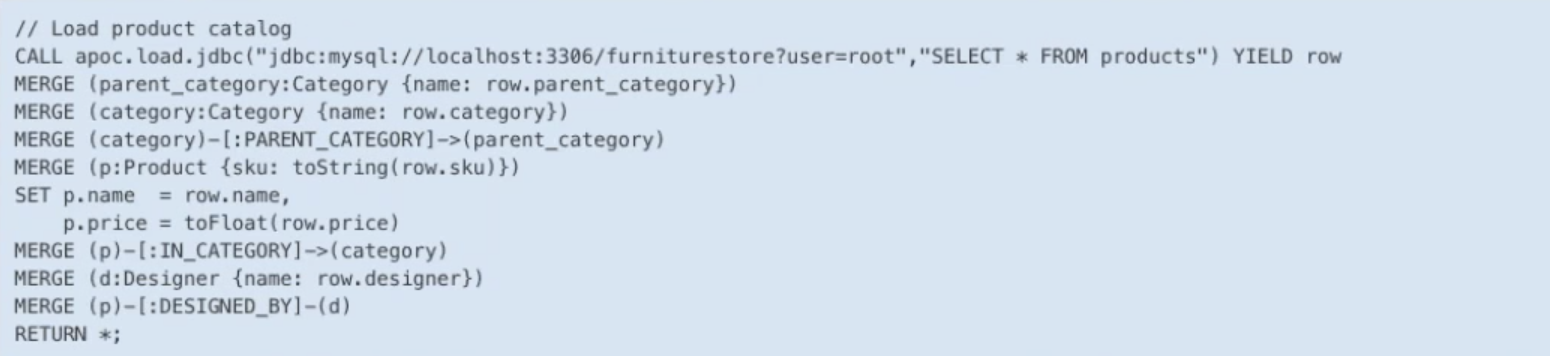
Designer <- Product -> Category.

1. **Collaborative filtering: Taking data from Customer and transaction silo:**

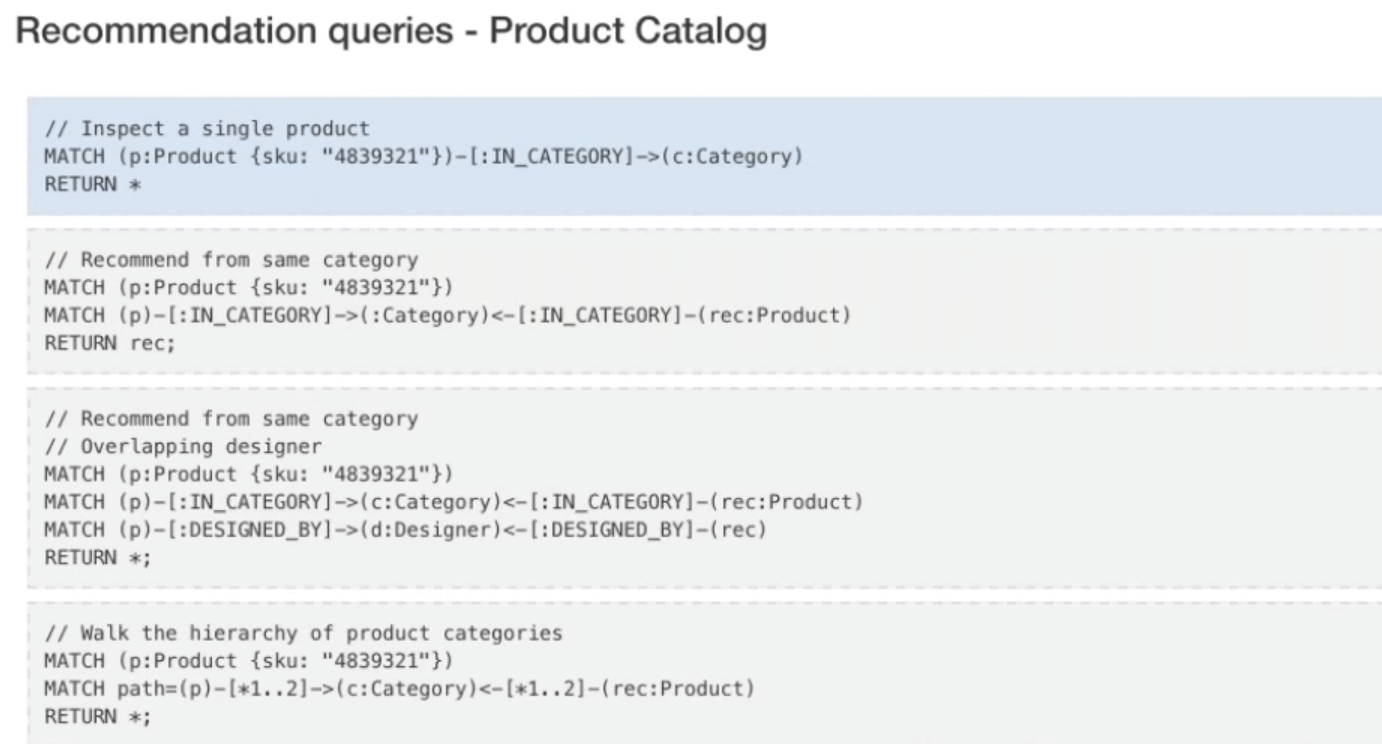
Include customers who have placed some orders and may be live in certain city. So now the schema has:



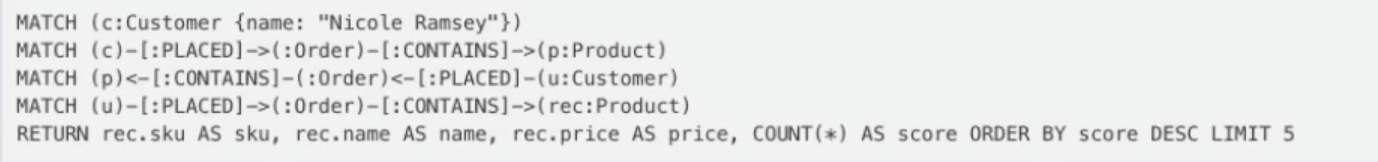
Product Silo:



Recommendation queries: Product Catalog:



Based on above created graph schema, let us create Collaborative filtering: Considering the products the specific user has purchased plus the products the other users have purchased that may fall into the same category.



Extension:

Making recommendations based on reviews.