Index Report - Similar Products Page

Candidate Indices
CREATE INDEX index_users_name ON users(name); - 9.96
CREATE INDEX index_products_name ON products(name); - 9.46
CREATE INDEX index_orders_userid ON orders(user_id); - 9.86

Beneficial Indices
CREATE INDEX index_products_name ON products(name); - 9.46

Index_products_name was beneficial because the name attribute in the products table is referenced across different tables our query. When performing cosine similarity, a product may have multiple similar products, so creating an index on the product name would create a single pointer to multiple tuples that have the same product name.

CREATE INDEX index_orders_userid ON orders(user_id); - 9.34 (useful)

Index_orders_userid should be beneficial because the user_id attribute in the orders table would be repeated due to one user having multiple orders.

Note: Many of the indices we expect to improve performance do not result in any speedup. We suspect this is the case because our queries rarely have WHERE clauses. Instead, we nest queries and assign names to those subqueries without creating additional tables. Our conditional statements in our queries are performed on the results of those individual sub queries, so we cannot create an index on them.

Experimental Indices
CREATE INDEX index_users_name ON users(name); - 9.96 (not really useful)
CREATE INDEX index_orders_userid ON orders(user_id); - 9.86 (not really useful)
CREATE INDEX index_orders_userid ON orders(user_id); - 9.34 (useful)

Final Choice of Indices
CREATE INDEX index_products_name ON products(name);
CREATE INDEX index_orders_userid ON orders(user_id);