Big Data IU-S25 Assignment 2 Report

Dmitry Beresnev
Innopolis University
Innopolis, Russia
d.beresnev@innopolis.university

***Abstract*—It is a report for the Assignment 2 of Big Data course taught at Innopolis University, Russia in Spring semester 2025.**

I. Introduction

A global e-commerce company, faces challenges managing exponential data growth from customer transactions, product catalogs, social networks, and inventory systems. The company wants to find an efficient way to store and analyze the data. They want to see which data model they can use to have fast data retrieval and query processing.

The company is struggling with three big data issues:

- **Volume**: large amounts of data per day
- **Variety**: Social network data, user behavior events, and campaign messages
- **Value**: Analyzing customer social networks for targeted marketing

The primary objective of this assignment is to design a big data solution to store petabytes (the provided dataset is only a snapshot) of raw data using effective, optimal and scalable data models that can handle growth and complexity without compromising quality, efficiency, or usability. The optimal design will enable analytics and transactions for real-time and batch data processing. Given task is to find the best data model for storing the big data and obtaining valuable insights from it.

II. Data Modeling

You can find the e-commerce dataset description in the assignment description from 'reference' directory. Three data models, each for specific database, were designed: Relational (for PostgreSQL), Document (for MongoDB) and Graph (for Neo4j). All the models are designed using Hackolade.

A. Relational Data Model

The first database version is modeled for PostgreSQL. There are six tables in the relational model, shown in Figure 1: 'products', 'events', 'campaigns', 'friends', 'clients' and 'messages'. This design has the following main features, some of which are inspired by notion of normalization:

- Clients are placed in separated table, what removes data redundancy in 'message' table. Here i assume that email provider is uniquely defined by client, which lines with dataset description and collected insights
- Products with related fields are separated into new table to remove data redundancy in 'events' table
- In table 'campaigns', campaign id ad campaign type are used as primary key, as how is stated in dataset description, campaign id is unique ony within one campaign type. This composite primary key produces corresponding composite foreign key in table 'messages', where message type corresponds to campaign type. Once again, this decision does not conflict the dataset description and collected insights, as both types have same set of values
- Note, that 'event' table, like 'products', also has price field. That is because event, like view or purchase, could occur when the price of product was different from the current product price. This feature is also introduced in other models

Other relations, together with data types and foreign and primary keys, shown in Figure 1 were deduced from dataset description and are quite natural, so there is no need to describe them in details.

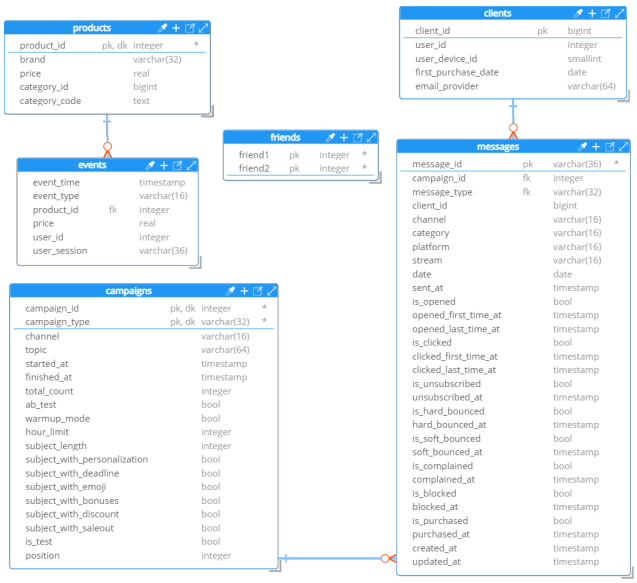

Fig. 1. Relational Tables for E-commerce

B. Document Data Model

The second database version is modeled for MongoDB. It creates a unique id by default automatically for each document whenever it is inserted into the database, so there was no need to set special 'ObjectID' type to custom identificators.

The main feature of document-based databases is that the data is represented in semi-structured documents: so it is possible to omit some fields if necessary and easily introduce arrays. I have decided to use these feature in the second design. Figure 2 shows five collections: 'users', 'clients', 'messages', 'campaigns', 'products'. Main features of proposed design are the following:

- Clients and products are placed in separated tables, as in relational model for same reasons
- Document-based model allows to use nested structures, so I decided to combine several fields into sub-documents based on their meaning. For example, fields related to subject within table 'campaigns' were combined into field 'subject'. Such design produce of more natural and intuitive way of data representation and subsequent querying
- Tables 'friends' and 'events' from relational model were transformed into collection 'users', as both tables refers to user id and describe the data related to the specific user: namely his/her friends and events occurred. So the element of collection 'users' stores the list of friends and events for specific user

Other data types and design elements shown in Figure 2 were deduced from dataset description and are quite natural, so there is no need to describe them in details.

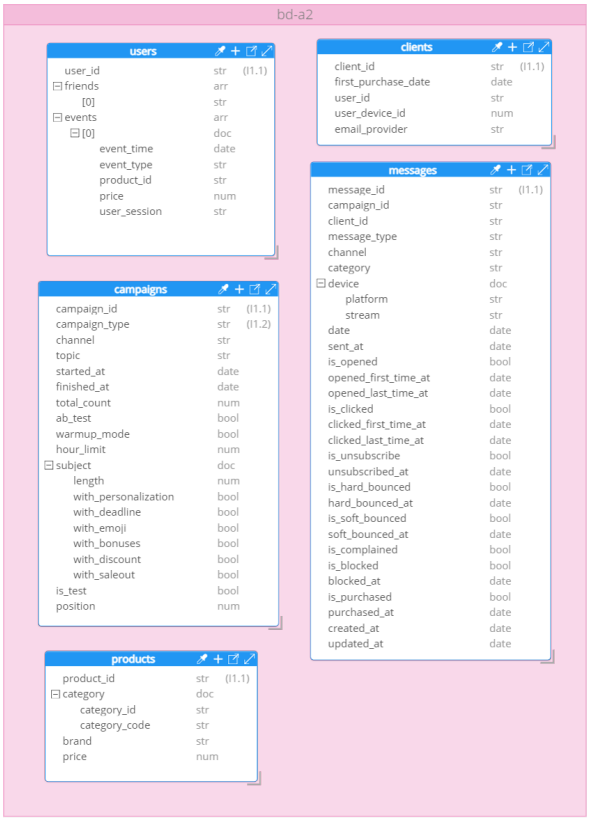

Fig. 2. Document Model for E-commerce

C. Graph Data Model

The third database version is modeled for Neo4j.

In a graph model, users and products are represented with nodes, and relationships with edges. Figure 3 shows the five types of nodes: 'Product', 'User', 'Client', 'Message' and 'Campaign'. There also are also seven tpe of relationships: 'VIEWED', 'ADDED TO CART', 'PURCHASED' (between user and product), 'FRIENDS WITH' (between users), 'BELONGS TO' (between client and user), 'RECEIVED' (between client and message) and 'SENT FROM' (from message to campaign). Properties of nodes and relations are listed in Figure 4. Main features of proposed design are the following:

- Clients and products are placed in separated tables, as in relational and document models for same reasons
- Table 'events' were transformed to three relationship types: 'VIEWED', 'ADDED TO CART', 'PURCHASED', and the corresponding data transferred to relation properties. The transformation is seemed very natural, as event, in the proposed sense, is indeed a relationships between user and product
- Note that the only bi-directional relationship is 'FRIENDS WITH', as design of dataset does not imply one-directional friendship. Other relations has naturall hierarchical meaning

Other data types and design elements shown in Figures 3 and 4 were deduced from dataset description and are quite natural, so there is no need to describe them in details.

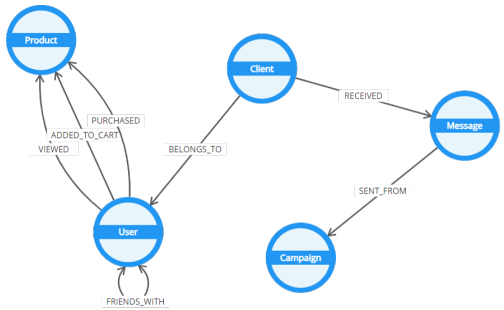

Fig. 3. Graph Nodes and Relations for E-commerce

ACKNOWLEDGMENT

The style of this report is inspired by the [1].

REFERENCES

[1] Serhat Uzunbayir. Relational database and nosql inspections using mongodb and neo4j on a big data application. In 2022 7th International Conference on Computer Science and Engineering (UBMK), pages 148–153, 2022. doi:10.1109/UBMK55850.2022.9919589.

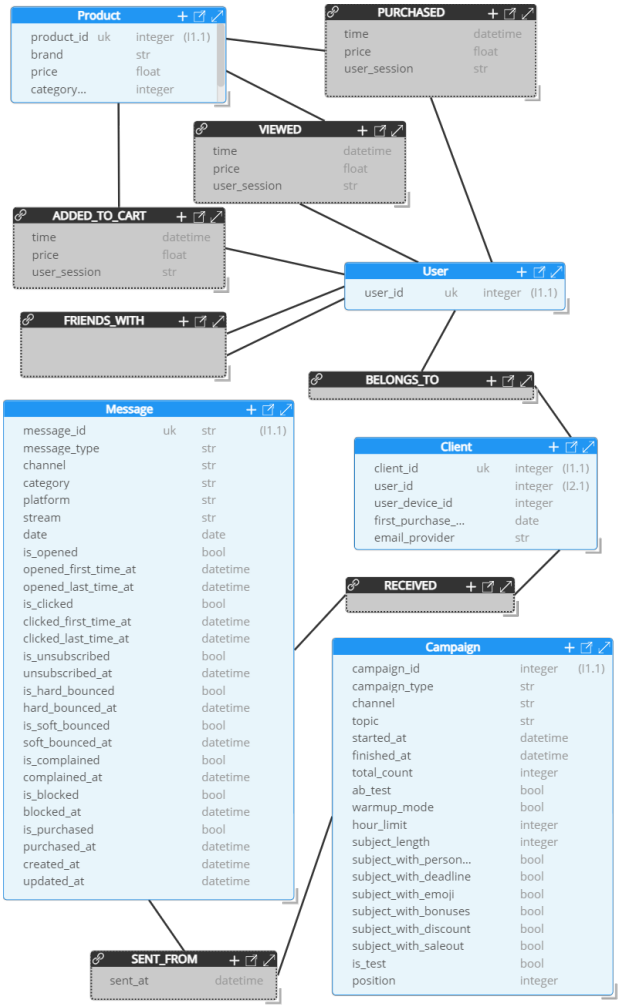

Fig. 4. Graph Model for E-commerce