E-Commerce Database Architecture Project

Project Story:

Sarah, a tech, fashion, and home décor enthusiast, eagerly logs into her favorite e-commerce platform and begins her shopping journey on her tablet, she had been researching a new pair of headphones on the platform. When she clicks on the headphones category, her recently viewed items appear instantly. The headphones are showcased with detailed tech specifications such as battery life, connectivity type, and weight, making her decision-making process straightforward. Seeing the real-time stock counter indicating limited availability, she quickly adds them to her cart.

Before she can proceed to checkout, she's distracted by a phone call. Hours later, she remembers her incomplete shopping cart. But this time, she's on her laptop. As she logs into the e-commerce site, she's pleasantly surprised. The platform immediately recognizes her, and not just by her login credentials. Her cart, still holding the headphones, waits for her exactly as she left it. The system's robust session management capabilities ensure that Sarah's shopping journey is uninterrupted and consistent, no matter the device.

Next, her interest drifts towards the fashion section where a gorgeous summer dress catches her eye. The platform lists the dress with attributes distinct from the headphones, such as size, material, and color options. The real-time size availability feature informs her at once which sizes are in stock. Entranced by the aqua-blue variant, she promptly adds the dress to her cart.

As she's about to checkout, a beautifully designed ceramic vase in the home décor section grabs her attention. This product introduces yet another set of attributes: dimensions, material (ceramic type), weight, and care instructions. The platform cleverly provides a visualization tool, letting Sarah gauge the vase's size against common household items. Convinced it would be perfect for her living room, she adds it to her cart.

With her selections made, Sarah moves to her cart. Here, she can review her products, adjust quantities, or remove items. The platform provides a breakdown of costs, including potential taxes and shipping fees. As she proceeds to the checkout, she's given various shipping options, ranging from standard delivery to expedited overnight shipping. She chooses a mid-tier option, balancing speed and cost.

The payment gateway offers multiple payment methods such as credit card or bank accounts. Sarah opts to pay with her credit card. With payment approved, she receives an immediate order confirmation, along with expected shipping and delivery dates.

However, when her items arrive, Sarah realizes the dress is a size too small. The platform simplifies the return process for her. It provides a step-by-step guide on packing the dress, printing the return label, and choosing return shipping options. Once her return is initiated, the platform keeps her informed about the return status and when to expect her refund or exchange.

Database Design:

As aspiring database architects, your task is to design the software backend of the e-commerce platform mentioned in Sarah's story. This must be done using a combination of both relational and non-relational database solutions.

1. Based on different functionalities and features described in the narrative, decide which database type (relational or non-relational) would be most appropriate for each function. Explain your reasoning for each function.
2. Draw an ERD for the relational database part of your design. Include tables, fields, types, and relationships.
3. Consider the unique attributes of each product category (e.g., headphones, dresses, vases) and how you would handle the diverse range of attributes across products.
4. Consider session management and how you would handle a user's session data across different devices and shopping sessions.
5. Identify and justify one part of your schema that can be denormalized for performance. Explain the read/write, consistency, and storage trade-offs this introduces.
6. Design a short data flow diagram showing how your relational and non-relational stores exchange data. Briefly describe data freshness/latency expectations and failure fallbacks.
7. Address potential challenges or concerns with your design and how you would mitigate them.
8. Which data or interactions would benefit from in-memory storage? How would you keep Redis data consistent or periodically synchronized with your main databases?
9. Given the richness of Sarah's shopping experience, it's evident that there's potential value in tracking user interactions on the platform. These interactions can include product views, search queries, clicks on products, time spent on particular product pages. Your task is to architect a database that not only supports e-commerce

functionalities but also effectively captures user behavior. Discuss your strategy and architecture for storing such behavior data.

10. Discuss the kinds of queries that a graph database could answer more efficiently than relational joins. Design a graph model showing nodes and relationships.

Fetching Data

Write a query or describe the mechanism to fetch the requested data. Some tasks may require multiple database stores.

1. Retrieve all products in the "fashion" category along with their associated attributes such as size, color, and material.
2. Retrieve the last five products viewed by Sarah within the past six months, ordered by most recent activity.
3. Check the current stock level for all items and return only items that are low in stock (e.g., less than five items).
4. Retrieve all products in the "fashion" category that are available in either blue color or large size.
5. Display the number of times each product page has been viewed, ordered by popularity (e.g., number of views).
6. Retrieve all recent search terms used by the user and categorize them based on frequency and time of day.
7. Fetch carts information such as device type (e.g., laptop, tablet), the number of items in the cart, and total amount.
8. Retrieve all orders placed by Sarah, showing order IDs, item details, payment methods, shipping options chosen, and the status of each order.
9. List all items returned by the user, along with the refund status, amount, and any restocking fees.
10. Retrieve the average number of days between purchases for Sarah.
11. Calculate the percentage of carts that did not convert to orders in the past 30 days.
12. Find the top 3 products most frequently purchased together with “headphones”.
13. For each user, compute days since last purchase and total order count.

Data Generation & Performance Evaluation

To test your schema and queries, generate synthetic data for your e-commerce platform using Generative AI or data simulation tools (e.g., ChatGPT, Gemini, or your own script).

Your generated dataset should:

- Represent realistic e-commerce scale (minimum: 1000 users, 5,000 products, 100,000 orders, 500,000 user events).
- Include product diversity (categories, attributes), sessions, and behavioral logs (views, searches, carts).
- Maintain referential consistency between relational and non-relational stores.

After generating data, execute or simulate all 13 queries above and evaluate query performance. Each query should run within 2 seconds for small-scale data (≤100K records per collection/table). If a query exceeds this limit, you must Identify the performance bottleneck, and propose or implement optimizations.

Submission Guidelines to Canvas:

- This is group project, only 2 students can collaborate on this project. Each student's specific contribution to the project must be clearly delineated. A brief description detailing what each student worked on will suffice.

- As part of your submission, include a separate revision log. This should summarize each change made to design decisions throughout the project. At least 5 distinct <u>major</u> revisions are expected. Each revision entry must:
    - Specify the design component being revised.
    - Detail the change made.
    - Provide a rationale for the revision.
    - Include the date of the revision.

- If generative AI tools are used to brainstorm or guide any design decisions, it is required to cite these interactions explicitly in your report. Alongside each citation, include a brief assessment or reflection on the AI's suggestions, highlighting your agreement, disagreement, or adaptations.

- Your report should be structured well to include all details and supplementary materials within the body of the report where relevant. Ensure your report provides a clear rationale for each design choice you made.