E-Commerce Recommendation Engine:

Leveraging SQL for Personalized Shopping Experiences

Problem Statement:

- An e-commerce platform wants to enhance its user experience and optimize product recommendations based on user interactions and past purchase history.

- The platform seeks to identify patterns in user behavior, such as popular products, frequently interacted categories, and purchasing habits, to provide personalized recommendations and improve customer engagement.

Solution Overview **-** SQL Implementation for E-commerce Recommendation Engine:

Data Modeling and Database Design:

- Design and create tables to store user interactions, product details, and past purchases.
- Define appropriate data types, primary keys, and foreign key constraints to ensure data integrity.
- Normalize the database schema to minimize redundancy and optimize query performance.

Data Analysis and Exploration:

- Utilize SQL queries to perform exploratory data analysis (EDA) on the database tables.
- Analyze user interactions by counting views, clicks, and purchases for each product.
- Identify popular products, frequently interacted categories, and user preferences through SQL aggregation functions and group by clauses.

User Segmentation and Profiling:

- Segment users based on their interaction history and purchasing behavior using SQL queries.
- Group users into clusters using techniques such as k-means clustering or hierarchical clustering.
- Create user profiles by aggregating user data, including demographics, preferences, and purchase history.

Databases Schema

Product Table:

Columns:

- **product_id:** Unique identifier for each product.
- **product_name:** Name of the product.
- **category:** Category to which the product belongs.
- **price:** Price of the product.
- **brand:** Brand of the product.

Interactions Table:

Columns:

- **interaction_id:** Unique identifier for each interaction.
- **user_id**: Unique identifier for each user.
- **product_id:** Unique identifier for each product.
- **interaction_type:** Type of interaction (e.g., view, add to cart, purchase).
- **timestamp:** Timestamp when the interaction occurred.

.

Past Purchases Table:

Columns:

- **purchase_id:** Unique identifier for each purchase.
- **user_id:** Unique identifier for each user.
- **product_id:** Unique identifier for each product that was purchased.
- **purchase_date:** Date when the purchase occurred.

Dataset Link to download

Product Table Data: Click here

Past Purchase Table Data: Click here

Interactions Table Data: Click here

Note: Add More Data and Table based upon your requirements

Complete SQL Analysis (Basics to Advanced)

1. Select all records from the Products table
2. Filter products by category 'Electronics'
3. Sort products by price in descending order
4. Count the number of interactions
5. Calculate the total purchase amount for each user
6. Retrieve the oldest purchase date
7. Join Products and Interactions to get product details with interaction type
8. Subquery to find products with more than 10 interactions
9. Update product price for a specific product
10. Delete an interaction record
11. Retrieve the top 5 users with the highest total purchase amount
12. Count the number of unique brands in the Products table
13. Window function to rank products by price within each category
14. Common Table Expression (CTE) to find the average price of products
15. Create an index on the user_id column of the Past_Purchases table
16. Retrieve the product with the highest total purchase amount
17. Create a view to show interactions with product details
18. Rollback a transaction if an error occurs while updating interactions
19. Count the number of interactions per product
20. List top N most popular products based on interactions
21. Retrieve product details along with user interactions
22. Find products with no interactions
23. Rank products by price within each category using window functions
24. Calculate the cumulative sum of total purchases by user
25. Find products purchased more than once
26. Retrieve interactions for products with prices above the average price
27. Create a CTE to calculate average product price by category
28. Use a CTE to find the top 3 users with the highest total purchase amounts
29. Calculate the percentage contribution of each product to the total sales amount
30. Identify users who made purchases of more than $500 in a single transaction
31. Calculate the average time between consecutive purchases for each user
32. Identify products that have been interacted with but not purchased
33. Find users who made purchases in the first and last quarter of the year
34. Calculate the average quantity of products purchased by users who interacted with products priced above the average price
35. Find users who have interacted with products across multiple categories