**DB-Design Assignment**

**Answer to Question No.1 is submitted separately in another folder.**

**Q.2- Explain about searching performance. How will you handle replication in SQL for searching & Reporting?**

In the E-Commerce application, the performance of searching plays an important role in providing a smooth user experience.

It can be achieved through followings methods-

**Denormalization-**

Denormalization is related to reducing the complexity of database so that searching performance of the data would be optimised. It involves duplicating data across tables to reduce the number of joins required for querying.

I had used the denormalization concept in my database design like-

**Denormalize Product Details-** In the Inventory module, I had stored all the relevant information related product details in a single table Instead of normalizing the product attributes into separate tables (e.g., product image, price, discount) from which I eliminate the need for joins when fetching product details, which improves mine search performance.

**Denormalize Order details-** Instead of keeping order status into separate table I prefer to add the status columns into same table so that it can simplify the querying of fetching the user order items and their status.

**Denormalize User details-** For Authentication & Authorization I had included the isVerified column in the same User Table.

**Indexing:**

Indexing is a technique used to improve the performance of database queries by creating data structures (indexes) that allow for quick look-up and retrieval of data.

I have implemented the indexes on various attributes-

**Indexes on product attributes-** I had applied the indexes on price attributes of the product table to get the product according to their price range.

**Indexes on Order module**- I had applied the indexes on order-date attributes of the Order table to get the details and order in between some dates to optimize the searching performance.

**Index on Notification Subscriptions**: In the Notification table, I had created index on the userID from which it improves the efficiency of retrieving and filtering subscriptions for sending the notifications.

**Database Replication**:

Implementing database replication helps in distributing the database workload across multiple servers. This can be achieved through techniques like – master-slave replication or multi-master replication.

**Q. 3- Explain what major factors are taken into consideration for performance.**

Following are the major factors are taken into consideration for performance -

**Caching**:

Implementing a caching layer, such as a distributed cache or in-memory cache, can improve searching performance by storing the frequently accessed data in memory. This helps in reducing the number of database queries for commonly requested data and resulting in faster response times.

**Load Balancing**:

Distributing the search and reporting requests across multiple database servers using load balancing techniques ensures efficient utilization of resources and helps handle increased user load.

Load balancing can be implemented using-

Hardware load balancers

Software load balancers

Cloud-based load balancing services.

**Query Optimization**:

Query Optimization used for searching and reporting can have a significant impact on performance. Techniques like query rewriting, query caching and optimizing database schema design can help improve query execution times.

**Denormalization and Materialized Views:**

Denormalizing the database schema and creating materialized views for frequently accessed data can enhance search and reporting performance. It involves duplicating and pre-computing data to eliminate the need for complex joins and aggregations during query execution, resulting in faster response times.

**Q.4- Mention about Indexing, Normalization and Denormalization.**

**Indexing**:

Indexing is a way to make database searches faster. It does this by creating a special data structure called indexes. Theses indexes help to quickly find and get the data we needed from the database. By creating indexes on columns, the database can locate the required data more efficiently, reducing the search time and improving performance.

**Indexes on product attributes**:

I had applied the indexes on price attributes of the product table to get the product according to their price range.

**Indexes on Order module**:

I had applied the indexes on order-date attributes of the Order table to get the details and order in between some dates to optimize the searching performance.

**Index on Notification Subscriptions**:

In the Notification table, I had created index on the user ID from which it improves the efficiency of retrieving and filtering subscriptions for sending notifications.

**Normalization**:

Normalization is a process of organizing data in a database to eliminate redundancy and improve data integrity. It involves breaking down the data into multiple tables and defining relationships between them. Normalization helps in reducing data duplication and ensures data consistency.

**First Normal Form (1NF):**

In 1NF each attribute contains only atomic values, and there are no repeating groups or arrays. This step typically involves breaking down composite attributes into individual atomic attributes.

In my ER diagram of product table attributes such as price, discount, specification I had kept into separate columns in the product table that denotes atomic value into a every column.

**Second Normal Form (2NF):**

In 2NF, each non-key attribute depends on all primary key and are not just part of it. If there are any partial dependencies, move the affected attributes to a separate table.

In my ER diagram I had created the separate table for cart, which includes attributes such as productID, quantity, and orderid. This table will have a foreign key referencing the order and product tables.

**Third Normal Form (3NF):**

In3NF, we ensure that there are no transitive dependencies, where an attribute depends on another non-key attribute. If such dependencies exist, move the affected attributes to a separate table.

In my ER diagram I had create a separate table for Buyer and Seller details, including attributes such as name, email. In my User table I declared a Enum which stores Buyer and Seller as value during user registration user must declare himself whether he/she was Buyer or Seller so here is the reference attributes in the form of Role attribute in user table which references the Buyer and Seller id.

**Denormalization**:

Denormalization is somethings related to reducing the complexity of database so that searching performance of the data would be optimised. It involves duplicating data across tables to reduce the number of joins required for querying.

I had used the denormalization concept in my database design like-

**Denormalize Product Details**:

In the Inventory module, I had stored all the relevant information related product details in a single table Instead of normalizing the product attributes into separate tables (e.g., product image, price, discount) from which I eliminate the need for joins when fetching product details, which improves mine search performance.

**Denormalize Order details**:

Instead of keeping order status into separate table I prefer to add the status columns into same table. This can simplify the querying of fetching the user order items and their status.

**Denormalize User details**:

For Authentication & Authorization I had included the isVerified column in the same User Table.

**Q. 5- How will you handle scaling, if required at any point of time.**

To handle scaling, we consider the following approaches-

**Database Replication**:

We can Implement database replication to have multiple copies of our database server. We can Configure replication mechanisms like master-slave or master-master replication depending on the requirements.

**Database Sharding**:

We can Implement database sharding techniques to horizontally partition our data across multiple database instances.

We can Split the data based on a certain criterion, such as customer IDs, product categories, or geographical regions of customers or users.

And after that Distribute the data shards across different database servers to distribute the load and improve scalability.

**Caching**:

We can Implement caching mechanisms to store frequently accessed data, such as product information or user sessions, in memory.

We can utilize caching solutions like Redis or Hazel cast to improve performance and reduce database load.

**Database Partitioning**:

We can partition the database such way that authorization and authentication handling modules shares different database and product, order and card module in different databases.

Partitioning can help distribute the data across multiple storage devices or servers, improving performance and scalability.

**Asynchronous Processing**:

We can utilize messaging systems like RabbitMQ or Apache Kafka for asynchronous processing.

**Content Delivery Network (CDN):**

We can utilize a CDN to cache and serve static content, such as product images, to reduce server load and improve response time.

**Containerization and Orchestration:**

We can Containerize the application using technologies like Docker.

Use container orchestration platforms like Kubernetes to manage and scale your application containers.

**Ques 6- Mention all the assumptions you are taking for solutions.**

There are few assumptions that can be taken-

**Authentication and Authorization**:

User authentication will be implemented using email/password or if user forgot or want to reset the password then users phone number required to get the OTP for the purpose of changing the password.

**Order Processing**:

After adding the product into Cart through Buyers, Sellers have full authority to reject or accept the order.

Order details, including order status and shipment information, will be stored, and made accessible to buyers and sellers.

**Notification System**:

A notification module will be implemented to send notifications to buyers regarding product availability, order status updates.

**Product Inventory Management**:

Sellers will have the ability to add new products to the inventory with details such as price, discount, specification.

**Forget Password Feature**:

The application will provide a mechanism for users to recover their forgotten passwords.

Password reset functionality will be implemented using secure procedures through sending Otp on registered phone number.