Question 5:

The response time of a query to database consist of several stages

A) Time required to establish connection
B) Time required to send the query
C) Time required for the query to complete in database
    1. Time waiting in queue
    2. Time required for execution planning of query
    3. Time required for execution of query
D) Time required to receive data from database
E) Time required to process the data by ORM
F) Time required to process the data in application
G) Time required for data to be transferred to user from application

Solutions:

- Partitioning, indexing , query optimization only effect section C of this process, usually this section alone is the bottleneck of response time we can always optimize it even more.
- Using a caching system like Hibernate second level cache is usually the second place we should look at because when data is available in cache it eliminates A,B,C,D all together and only add cache lookup time instead of them. However given the description of the problem which describe the data size as huge and data access pattern as random this solution will not help in this case since for most of data access chance of data being in cache is very low and by adding this we might actually experience a higher latency for our responses.
- A good database connection pool like Hikari which is being used by spring frame work will help greatly in reducing time of part A.
- Changing transaction commit patterns will help with part B and C, 100 update/read distributed among 100 transactions are much slower than 100 update/read in on transaction since we can help query planner and scheduler greatly in optimizing the actions we want to (if we can manage it since it requires change in software).
- Moving the database closer to the server by reducing latency and bandwidth will help with part D, However the greatest gain can be achieved by reducing amount of data that is being transferred by actually choosing fields instead of whole row of table (which complicate Data Access Layer in our software)
- Using a compressed transfer protocol like SSL Compression options can greatly reduces the size of data being transferred(no serious effect on software stack).
- ORM process time is negligible compared to the others but choosing a more efficient ORM will help as well, or by removing ORM completely. However a better approach is to use Native Queries to transfer most of computation into the database which will help reduce D.