**Q1.** An audit department generates and accesses the audit reports only twice in a financial year. The department uses AWS Step Functions to orchestrate the report creating process that has failover and retry scenarios built into the solution. The underlying data to create these audit reports is stored on S3, runs into hundreds of Terabytes and should be available with millisecond latency. As a solutions architect, which is the MOST cost-effective storage class that you would recommend to be used for this use-case?

a) Amazon S3 Standard

b) Amazon S3 Intelligent-Tiering (S3 Intelligent-Tiering).

c) Amazon S3 Standard-Infrequent Access (S3 Standard-IA).

d) Amazon S3 Glacier Deep Archive.

**Ans**: c) Amazon S3 Standard-Infrequent Access (S3 Standard-IA)

**Explanation**: Since the data is accessed only twice in a financial year but needs rapid access when required, the most cost-effective storage class for this use-case is S3 Standard-IA

**Q2.** The engineering team at a Spanish professional football club has built a notification system for its website using Amazon SNS notifications which are then handled by a Lambda function for end-user delivery. During the off-season, the notification systems need to handle about 100 requests per second. During the peak football season, the rate touches about 5000 requests per second and it is noticed that a significant number of the notifications are not being delivered to the end-users on the website.

As a solutions architect, which of the following would you suggest as the best possible solution to this issue?  
a) The engineering team needs to provision more servers running the Lambda service.

b) Amazon SNS has hit a scalability limit, so the team needs to contact AWS support to raise the account limit.

c) The engineering team needs to provision more servers running the SNS service.

d) Amazon SNS message deliveries to AWS Lambda have crossed the account concurrency quota for Lambda, so the team needs to contact AWS support to raise the account limit.

**Ans**. d) Amazon SNS message deliveries to AWS Lambda have crossed the account concurrency quota for Lambda, so the team needs to contact AWS support to raise the account limit.

Correct option:

Amazon SNS message deliveries to AWS Lambda have crossed the account concurrency quota for Lambda, so the team needs to contact AWS support to raise the account limit  
Amazon Simple Notification Service (SNS) is a highly available, durable, secure, fully managed pub/sub messaging service that enables you to decouple microservices, distributed systems, and serverless applications.

With AWS Lambda, you can run code without provisioning or managing servers. You pay only for the compute time that you consume—there’s no charge when your code isn’t running.  
AWS Lambda currently supports 1000 concurrent executions per AWS account per region. If your Amazon SNS message deliveries to AWS Lambda contribute to crossing these concurrency quotas, your Amazon SNS message deliveries will be throttled. You need to contact AWS support to raise the account limit. Therefore this option is correct.  
Incorrect options:

Amazon SNS has hit a scalability limit, so the team needs to contact AWS support to raise the account limit – Amazon SNS leverages the proven AWS cloud to dynamically scale with your application. You don’t need to contact AWS support, as SNS is a fully managed service, taking care of the heavy lifting related to capacity planning, provisioning, monitoring, and patching. Therefore, this option is incorrect.

The engineering team needs to provision more servers running the SNS service  
The engineering team needs to provision more servers running the Lambda service  
As both Lambda and SNS are serverless and fully managed services, the engineering team cannot provision more servers. Both of these options are incorrect.

**Q3.** A retail company has developed a REST API which is deployed in an Auto Scaling group behind an Application Load Balancer. The API stores the user data in DynamoDB and any static content, such as images, are served via S3. On analysing the usage trends, it is found that 90% of the read requests are for commonly accessed data across all users.

As a Solutions Architect, which of the following would you suggest as the MOST efficient solution to improve the application performance?

1. Enable ElastiCache Redis for DynamoDB and CloudFront for S3
2. Enable DynamoDB Accelerator (DAX) for DynamoDB and CloudFront for S3
3. Enable DAX for DynamoDB and ElastiCache Memcached for S3
4. Enable ElasticCache Redis for DynamoDB and ElastiCache Memcached for S3

**Ans.** Enable DynamoDB Accelerator (DAX) for DynamoDB and CloudFront for S3

Q4