Domain 2: Define Performant Architectures

1. D. There is no way to reason through this; it is a matter of memorization. There is no charge associated with data replication in this scenario.

2. C, D. All of these are valid options. Although it's not particularly common, you can set up a read replica in an on-premises instance. Additionally, read replicas are often created in separate regions from the primary instance, to improve performance for clients closer to different regions than the primary instance.

3. C. A read replica configuration is aimed squarely at increasing database performance, specifically the performance of reading data from an RDS instance.

4. D. All three of these databases support read replicas. Most other databases supported by RDS (Oracle, for example, or Aurora) offer other approaches to gain similar functionality to read replicas but do not support the AWS read replica functionality.

5. B. Currently, read replicas in RDS are only supported by MariaDB, MySQL, and PostgreSQL.

6. A, C. A read replica is a read-only instance of a database created from a snapshot of the primary instance (A). Read replicas can be in the same instance, or a different one, as the primary instance (so B is false). Read replicas are updated via asynchronous replication—the most performant approach—from the primary database.

7. C, D. Read replicas can be in a different region than the primary instance (D), and they replicate all the databases in the primary instance (C). You can have up to five read replicas at a time for a single instance (so A is false). While MySQL and MariaDB are supported (B), Aurora is not.

8. D. The root issue here is that a read replica setup only allows for five read replicas. This is not a limit that can be raised by AWS either (so C is out). Option A won't address the issue, and option B isn't accurate; there are no EU limitations affecting the issue here. The only answer that would result in being able to create the instance is D: By turning off an existing instance, you can create a new fifth replica in the desired region.

9. C, D. Read replicas are focused on performance, so you can generally eliminate any answers related to disaster recovery—in this case, A. Read replicas work with RDS databases, as well, so B is out; on-premises databases aren't supported. This leaves C and D, which are both valid.

10. B, C. No backups are taken from any instance automatically, including the primary instance, so A is false. Since each read replica has its own database instance running, both B and C are valid. Replication is asynchronous rather than synchronous (so D is false).

11. A, B. A is false because you can create read replicas in the same AZ as the primary instance. There is no requirement to use multiple AZs, as there is with a Multi-AZ setup. B is also false; read replicas provide no disaster recovery options. Both C and D are true.

12. A. Only A is correct. A Multi-AZ setup is focused on disaster recovery and fault tolerance, while read replicas provide performance and scalability.

13. D. There is no difference in how applications communicate with read replicas as compared to the communication with non-replica instances. In fact, applications don't "know" that they're communicating with a read replica other than an inability to make writes.

14. A, D. A and D are both solutions that would be aided by additional read-only instances. B is not a valid answer because updating records would still only be possible with the primary instance; read replicas don't support writes. C is incorrect because read replicas do not provide automated fault recovery.

15. B. You need to be careful here. While read replicas are not advertised or even suggested as solutions for disaster recovery, option B does provide a somewhat manual process to use them in that manner. While you get no automated backups or failover (A or C), you can manually promote a read replica instance to a stand-alone instance if you have to. Still, a Multi-AZ setup is almost always a more robust solution for fault tolerance.

16. A, B. Both A and B are ideal situations for read replicas. C is the usual incorrect answer: read replicas don't provide automated backups. And D is not accurate; the actual database processing doesn't improve; you are merely adding more sources for reading data for clients.

17. D. AWS does not support circular replication through RDS. While some of the databases supported by RDS do, RDS itself does not provide access to this functionality.

18. A, C. You can create a read replica through the AWS console (A), the AWS API (C), and the AWS CLI (not mentioned, but still true).

19. B. As has been said numerous times, read replicas are not a backup strategy, nor do they cause automatic backups to be set up. However, you must turn on automatic backups for the primary database instance to enable read replicas.

20. A. This bears careful reading. Amazon RDS does not support circular replication, which means one database reads from a second database but then is replicated back by that second database. However, it is absolutely permissible for one database to replicate another database and then be the source for a third database. This makes option A correct.

21. D. There is no difference in response to a change in the backup window based on how that window is changed (API, console, etc.). All changes take place immediately.

22. B. This is another straight memorization question: Amazon RDS backups can be retained for up to 35 days, and no longer.

23. C. There are two components to this question: using RDS or EC2 for Oracle hosting and the class of storage to select. While RDS is a better option in the general case, it is likely not possible to use RDS in this scenario due to the custom plug-in required. This eliminates A and B. Given an installation on EC2, then, the question becomes which storage class is faster: provisioned IOPS or magnetic. The answer here is always provisioned IOPS.

24. B, C. Option C should be the immediately obvious first choice. Anytime you have custom plug-ins, you will likely need to install your database on an EC2 instance rather than using RDS. Options A and D are really both about network routes and services around your database, and both can be accomplished without affecting your EC2 vs. RDS decision. This leaves B, which also logically makes sense: If you have a very large database, and it will grow (as almost all databases do), then sizing restraints on RDS can be a limiting factor.

25. A, C. This should be an easy question if you're prepared. While it's easy to forget if Aurora and MariaDB are RDS options—they are!—you should know that DynamoDB is AWS's NoSQL database, and Redshift is a data-warehousing solution.

26. A. This is not particularly difficult as long as you understand that a Multi-AZ deployment is concerned with failover, not performance. Option A is correct: There is no particular performance increase in a Multi-AZ deployment, unless read replicas are also turned on (which isn't specified). B is false because only the primary database responds to requests in a Multi-AZ deployment. C is actually a true statement but does not have a bearing on the subject of the question: performance. And D doesn't actually make sense in the context of the question at all!

27. A. This one is a little tricky as it requires understanding what default options AWS puts in place. By default, root volumes are terminated on instance deletion, and by default, additional EBS volumes attached to an instance are not. This makes option A true. However, note that these settings can be changed! Also note that option D is not true in any configuration.

28. B. The default for all new accounts is 100 allowed S3 buckets; this is consistent across AWS and does not change via configuration (meaning that C and D are not correct). However, this value can be raised through asking AWS for an exception and providing a reasonable justification, making B the correct answer.

29. B. Replication occurs synchronously from a primary instance to a secondary instance in a Multi-AZ setup. Asynchronous replication only occurs in a read replica setup (which can be enabled in addition to a Multi-AZ setup).

30. C. Replication occurs asynchronously from a primary instance to the various read replicas in a read replica setup. As a result, updates are not guaranteed to be instant on the read replicas. Synchronous replication occurs in a Multi-AZ setup.

31. A, B. Classic load balancers support both IPv4 and IPv6. They support HTTP/1 and HTTP/1.1, but only application load balancers support HTTP/2. Further, you must register individual instances, rather than target groups, with classic load balancers; registering target groups is a functionality only available with application load balancers.

32. A. AWS accounts allow you five elastic IP addresses per region by default. As with most AWS defaults, this can be raised by contacting AWS and providing a reasonable justification.

33. C. Officially, instances can have up to 28 attachments. One of those attachments is the network interface attachment, leaving 27 attachments available for EBS volumes. However, the better approach is to remember that an instance can attach to a root volume and several more volumes (more than two); this eliminates options A and B. Additionally, instances cannot have unlimited attachments. This leaves the correct answer, C.

34. A. Be careful with the wording, to ensure that you do not misread this as asking how many EBS volumes can be attached to an EC2 instance (a different question altogether). A single EBS volume can only be attached to one instance at a time.

35. B, C. This should be an easy answer: Application load balancers, as well as classic load balancers, only support HTTP and HTTPS.

36. A, C. RDS provides two (and only two) methods for backing up RDS databases at this point: automated backups and automated snapshots. S3 lifecycle management policies are not applicable to RDS databases, and data pipeline is not relevant in this context.

37. B. Data written to and from cache is ephemeral, and if your instance is reading and writing that data frequently, the only way to ensure that your snapshot isn't missing data is to stop the instance from running altogether and to then take a snapshot (B). Both A and C will take snapshots but will likely miss any cached data. With option D, you cannot detach a root volume from an instance (it's unclear from the question if the cached data is being written to EBS, EFS, or another storage mechanism in any case), and so it is not a safe choice.

38. B, C. Option A is invalid because Multi-AZ is a disaster recovery solution; the primary database is the only instance that can respond to traffic in normal operation (unless read replicas are also set up). Option B is valid; caching user data would reduce round trips to the database and should reduce lag for users. Option C also makes sense, as having additional databases from which to read should decrease network latency to a single RDS instance. Option D is not helpful as the problem appears to be in retrieving credentials, not in the web tier itself.

39. C. Standard S3 (A) is not a bad choice, but is the most expensive, and both it and S3-IA (B) are more expensive than S3 One Zone-IA because of their increased availability and resilience. The key here is that photos can be lost without an issue, making S3 One Zone-IA the better option. S3 RRS is no longer recommended by AWS.

40. C. This should be automatic: Anytime a large data transfer is involved (especially on an AWS exam!), the answer should be Snowball. This comes up a lot and should be an easy correct answer.

41. A, D. The only tricky answer here is B. While Multipart Upload absolutely would improve the experience of uploading large files (larger than 10 GB, for example), it is not required; therefore, option B is not the best option to choose. Options A and D both are only possible with Multipart Upload enabled. Option C is false, as security is not related to Multipart Upload.

42. C. A placement group is concerned primarily with network throughput and reducing latency among EC2 instances within a single availability zone. AWS does support a placement group spanning multiple AZs via spread placement groups, but unless "spread" is specifically mentioned, you should assume the question references a "normal" (or "cluster") placement group.

43. A, B. Cluster placement groups (the default type of placement group) must be made up of instances that exist within a single availability zone (A). This results in increased throughput for network activity (B) but does not affect actual disk performance when writing to S3 (C). Instances can also be of different types, so D is also false.

44. B, C. Spread placement groups can span availability zones and support up to seven instances per zone (C). Like cluster groups, this results in increased throughput for network activity (B). You must specify the distinct underlying hardware for spread placement groups, which means that D is false.

45. D. This is a question where the answer is nonintuitive. All the S3 storage classes have the same durability. Even S3 One Zone-IA has 11 9s of durability in the single availability zone in which it resides.

46. A. Availability starts at 99.99% for S3 and then decreases to 99.9% for S3-IA, 99.5% for S3 One Zone-IA, and finally N/A for Glacier.

47. D. This question is easy if you recall that lifecycle transitions are concerned with moving between these storage classes. Therefore, all of these classes support those transitions.

48. A, C. All S3 and S3-IA data is stored in a single region and within at least three availability zones within that region. There is no "global" region for S3 storage.

49. C. Redshift is the only database or service in this list suitable for online analytics processing (OLAP). DynamoDB is an object database (NoSQL), and both Aurora and Oracle are relational databases, better suited for transaction processing.

50. B. All the major databases supported by RDS—MariaDB, SQL Server, MySQL, Oracle, and PostgreSQL—allow up to 16 TB of storage for a provisioned IOPS volume.

51. A. A provisioned IOPS EBS volume is a solid-state drive that provides the highest performance volume.

52. B. A cold HDD is the cheapest EBS option, so B is correct. It is not solid state (A), it is not appropriate for data warehousing (C), and it is not available to be used as a boot volume.

53. A, D. This is easiest to remember by noting that HDD types are not available to use as boot volumes. The SSD types (A, D) are, and are correct.

54. B, D. A General Purpose SSD is the less-expensive SSD (compared to provisioned IOPS), so B is a valid answer. It also provides low-latency performance and is bootable. Option A is more suitable for provisioned IOPS, and C is better for a throughput-optimized HDD.

55. A, B. Magnetic volumes are older and generally not used much. They are ideal for saving money (A) or for infrequently accessed data (B).

56. B, C. Provisioned IOPS volumes are not inexpensive (A) but are well-suited for critical database workloads and throughput (B and C).

57. A, C. An SSD volume is best for transactional workloads (A) with a large number of small I/O sized read/write operations.

58. B, D. An HDD-backed volume is best for streaming workloads where throughput needs to be maximized over IOPS.

59. C. While it is possible that a General Purpose SSD might be sufficient to support an Oracle installation that doesn't do a lot of processing, the best option is C, a provisioned IOPS SSD. Provisioned IOPS handles transaction processing well and will handle the large number of reads and writes that an Oracle installation would need.

60. A. This use case is one where access needs to be minimal, as does cost. If you have infrequently accessed data and cost is a major driver, magnetic drives might be a good option. While throughput-optimized HDDs are still cheaper than SSDs, magnetic is the cheapest option and would work fine for a set of data that is accessed without high performance needs.

61. A, C. You can boot an EC2 instance off any SSD type, as well as the magnetic type. HDD options are not available to use as boot volumes.

62. B, C. The HDD EBS volume types are not available to use as boot volumes, so B and C are the correct answers.

63. C. There is no such thing as a weighting load balancer. The other options are actual options.

64. A, C. An ELB is an elastic load balancer and generally refers to a classic load balancer. An ALB is an application load balancer. So A and C are valid; MLB and VLB are not acronyms or abbreviations for load balancers.

65. C. An ALB operates at Level 7, the individual request (application) level. Network load balancers operate at Level 4, the connection (transport) level. No load balancers operate at Level 1, and there is no Level 8 in the TCP/OSI stack.

66. B. An ALB operates at Level 7, the individual request (application) level. Network load balancers operate at Level 4, the connection (transport) level. No load balancers operate at Level 1, and there is no Level 8 in the TCP/OSI stack.

67. B, C. Classic load balancers operate at both the connection (Level 4) and the request (Level 7) layer of the TCP stack. An ALB operates at Level 7, the individual request level. Network load balancers operate at Level 4, the connection (transport) level.

68. D. With the newer features of an ALB, all of these use cases are supported. It is important to recognize that ALBs can balance across containers, making B true, and pointing you to D: all of the above.

69. B. This is a difficult question, and right at the edges of what the Architect exam might ask. However, it is possible to use a load balancer to operate within a VPC. It can be pointed internal, instead of Internet-facing, and distribute traffic to the private IPs of the VPC.

70. B. ALBs offer the most flexibility in routing and load distribution.

71. C. Network load balancers can handle the extremely high request load mentioned in the question as well as route between static IP addressed instances.

72. B. An ALB offers SSL termination and makes the SSL offload process very simple through tight integration with SSL processes. While an ELB will handle SSL termination, it does not offer the management features that ALBs do.

73. D. Both ALBs and ELBs offer SSL termination. While an ALB is a better choice when considering the management of SSL certificates—due to its ACM integration—both ELBs and ALBs are correct when considering just SSL termination.

74. D. Route 53 supports up to 50 domain names by default, but this limit can be raised if requested.

75. D. Route 53 supports all of the records mentioned, including alias records.

76. A. Route 53 does support zone apex (naked) domain records.

77. A, B. ElastiCache offers two engines: memcached and redis. Neither C nor D are even real things!

78. D. ElastiCache, when used through AWS, handles all of these tasks and more: hardware provisioning, software patching, setup, configuration, monitoring, failure recovery, and backups.

79. B, D. This is another example of an odd answer set, which sometimes appears on the AWS exam. In this case, all answers are valid, which means choosing two: B and D (D references the remaining two, A and C)!

80. A, C. ElastiCache is an in-memory data store (A) that shards across instances (C). It is not in itself a data distribution mechanism, which is why B is not correct. And it is not a monitoring solution at all (D).

81. D. CloudFront allows interaction via CloudFormation, the AWS CLI, the AWS console, the AWS CLI, the AWS APIs, and the various SDKs that AWS provides.

82. A, C. CloudFront can front a number of AWS services: AWS Shield, S3, ELBs (including ALBs), and EC2 instances.

83. B, C. CloudFront can front a number of AWS services: AWS Shield, S3, ELBs (including ALBs), and EC2 instances.

84. A, B. CloudFront can front a number of AWS services: AWS Shield, S3, ELBs (including ALBs), and EC2 instances. It also most recently supports Lambda@Edge as an origin.

85. A, C. This is a bit difficult, as CloudFront is typically associated with performance (A), and not a lot else. However, CloudFront also provides deep integration with many managed AWS services, such as S3, EC2, ELBs, and even Route 53.

86. B, D. CloudFront automatically provides AWS Shield (standard) to protect from DDoS, and it also can integrate with AWS WAF and AWS Shield advanced. These combine to secure content at the edge. HTTPS is not required (so A is incorrect), and there is no KMS involvement with CloudFront (C).

87. B, D. Edge locations number more than both regions and availability zones.

88. A, C. CloudFront is easy to set up and lets you create a global content delivery network without contracts (A). It's also a mechanism for distributing content at low latency (C). Creating websites and the actual file storage reference in B and D are not features of CloudFront but of LightSail (for example) and S3, respectively.

89. A, B. CloudFront can serve static content from S3 and dynamic content generated by EC2 instances.

90. B. When you create a CloudFront distribution, you register a domain name for your static and dynamic content. This domain should then be used by clients.

91. A, C. CloudFront will always handle requests that it receives. It will either return the requested content if cached (A) or retrieve that content by requesting it from an origin server (C). It will not redirect the client (D), nor will it pass the request on directly (B).

92. C, D. Both A and B are true. C is not, as routing will occur to the nearest edge location to the client, not the origin server. D is false; RDS is not a valid origin server for CloudFront.

93. D. There is no charge associated with data moving from any region to a CloudFront edge location.

94. A, B. S3 stores files and CloudFront stores copies of files, so A is true. Both also encrypt their files (B) as needed. Only CloudFront caches files (C), and only CloudFront can guarantee low-latency distribution (D).

95. B, D. CloudFront can store and serve both static (HTML and CSS, option D) and dynamic (PHP, option B) content. SQL queries cannot be directly returned, nor can an actual Lambda function. You can front the result of a Lambda@Edge function, but not the function itself.

96. A, B. CloudFront supports a variety of origin servers, including a non-AWS origin server (A). It supports EC2 (B), regardless of region, as well. It does not support RDS or SNS.

97. A. An edge location is a data center that delivers CloudFront content. Edge locations are spread across the world.

98. B. A distribution is the setup including your origin servers and how the content from those servers is distributed via CloudFront. It does not specifically refer to cached content at any given point in time.

99. D. Edge locations check for updated content every 24 hours by default, but this value can be changed.

100. A. Edge locations can be set to have a 0-second expiration period, which effectively means no caching occurs.

101. B, C. The most obvious culprit is a very low expiration period (B). Ensure that time is not close to 0. Beyond that, it's possible that—especially in conjunction with a very low expiration period—your compute resources or storage resources are getting flooded with requests. Consider using additional origin servers.

102. A. Setting an expiration period to 0 expires all content (A). It actually would slow down response time and has nothing to do with DDoS attacks. While it would technically always return the most current content (D), that's not a good reason to take this step; it defeats the purpose of CloudFront if the period is left at that value.

103. B, C. First, there is no mechanism either in the AWS console (A) or the AWS CLI (D) to interact directly with files on CloudFront distributions or edge locations. Second, the correct solution is to remove the file and then wait for the expiration period to cause a reload—which can be forced by setting that time to 0.

104. C. You need to remove a file from CloudFront's origin servers before doing anything else, because files are replicated from the origin server. If the file exists, it will end up on an edge location. Then, with the file removed, the expiration period can be set to 0, and the cache will be updated—resulting in the file being removed.

105. C. The invalidation API is the fastest way to remove a file or object, although it will typically incur additional cost.

106. A. S3 will always be the most available S3 storage class. This should be an easy correct answer.

107. D. This can be a bit tricky. While S3 is more available, all the S3-based storage classes provide the same first byte latency: milliseconds. Remember that performance is identical; availability is not.

108. D. This is another semi-trick question. With the exception of Glacier, retrieving data from the various S3 storage classes should be virtually identical (network issues notwithstanding). The classes differ in availability, but not in how fast data can be accessed. (They also differ in terms of charging for the number of accesses of that data.)

109. C. Glacier data retrieval, using the standard class, takes 3–5 hours on average.

110. B. The difference between S3 and S3-IA is cost, and frequency of access (B). Retrieval is just as fast as S3 (so A is wrong), and data in S3-IA is stored redundantly (C and D).

111. C, D. C and D are both false. RDS instances cannot be origin servers, and the default expiration period is 24 hours, not 12.

112. D. CloudFront allows all of these as origin servers: S3, EC2 instances, ALBs, etc.

113. D. A collection of edge locations is a distribution.

114. C. An RTMP distribution is the Adobe Real-Time Messaging Protocol and is suitable for using S3 buckets as an origin server to serve streaming media.

115. A, C. CloudFront supports both web distributions and RTMP distributions. Media and edge distributions are not valid distribution types.

116. A, B. You can read and write objects directly to an edge location. You cannot delete or update them directly; only the CloudFront service can handle that.

117. A, D. ElastiCache is ideal for high-performance and real-time processing as well as heavy-duty business intelligence. It does not shine as much with offline transactions, where speed is less essential, and it's not at all suitable for long-term or record storage.

118. B, D. Consider ElastiCache as only useful for storing transient data. Further, it's not a persistent store; therefore, it's great for caching data from a database or message service.

119. A, C. Consider ElastiCache as only useful for storing transient data. Further, it's not a persistent store; therefore, it's great for caching data from a message queue or providing very fast ephemeral storage.

120. A. ElastiCache uses shards as a grouping mechanism for individual redis nodes. So a single node is part of a shard, which in turn is part of a cluster (option A).

121. D. A storage gateway using cached volumes will cache frequently accessed data while storing the entire dataset on S3 in AWS.

122. C. A storage gateway using stored volumes will store all data locally while backing up the data to S3 in AWS as well.

123. C. A storage gateway using stored volumes will store all data locally, while all the other solutions store data in the cloud. Accessing local data will always be faster than accessing cloud data.

124. A. A storage gateway using stored volumes will store all data locally, providing low latency access to that data. Further, the entire dataset is backed up to S3 for disaster recovery. S3 is durable and available, but not as fast as accessing local data. A VTL provides a tape backup interface, but not necessarily fast data access.

125. B. The problem here is trying to tag individual folders. You can use IAM for permissions, but a particular folder cannot be tagged separately from other folders; only an entire bucket can be tagged.

126. D. A customer gateway with stored volumes provides the lowest latency for file access. A cached volume would not work because the majority of the files are the concern rather than just a small subset.

127. A. Configuring read replicas throughout all regions would provide the best response time on reads for customers spread across those same regions (A). While using multiple regions does provide some disaster recovery help, read replicas are really not a particularly effective disaster recovery approach. As for option C, read replicas do not increase network throughput; they just spread load out over the replicas, which may or may not desaturate the networks involved.

128. C. Read replicas are ultimately about providing faster read access to clients. If all your clients are in one region, then there is little advantage to adding read replicas to additional regions. Instead, providing replicas in the same region as the clients gives them the fastest access (C).

129. B. An argument could be made for option A here; customers will not be routed to a different region than the closest one if there are resources in a close region (so C is wrong) and D doesn't make much sense here. However, it is possible that if the primary region failed altogether, you could convert a replica in another region to a primary database, meaning that B has some merit.

130. A, D. Read replicas can be backed up manually, and of course read from. However, they effectively become read-only instances, so cannot be written to. You also cannot fail over to a read replica. You can convert it to a stand-alone instance, but that is a manual process that is not a failover. Therefore, A and D are correct.

131. A, D. This is a tough question because there is not much context other than knowing the database is not performing well, in a general sense. However, of the options given, switching to DynamoDB and adding Multi-AZ would do little to improve performance. (Note that switching to DynamoDB could help, as DynamoDB auto-scales to handle load, but this is still not the best of the available answers). Adding read replicas and looking at bigger instances are safer and better answers given this limited context.

132. C, D. Only C and D would have a guaranteed effect here. While it is possible that S3 would deliver the PDFs faster, you'd still have heavy network traffic over AWS, and there's no guarantee given the information here that S3 would be faster than RDS. B looks appealing, but note that the files are not accessed frequently. This means that caching is not going to help response time, as the files aren't accessed enough for caching to kick in and be effective. The best options are setting up read replicas and looking at beefing up the database instance.

133. B, C. There are typically a lot of "the database is being overwhelmed" questions on the exam, and this is one of those. The key here is understanding that data is accessed infrequently, meaning that caching solutions (A and D) likely won't help. Further, the staff is on-site, meaning that a customer gateway (C) could be a valid solution. Finally, it's almost always safe to at least consider upgrading the database instance.

134. B. Here, the key details are infrequent data access and a geographically distributed customer base. This means that read replicas spread out across the country are the best bet (B). Caching won't help, so A and D are out, and a storage gateway won't help customers that aren't accessing the data on-site.

135. D. This is a tough question, as several answers are valid. However, the key consideration here is that a single image is accessed several thousand times a day. Rather than adding instance power or read replicas (A and B), caching the images is the best approach, as it reduces overall database reads. In general, pulling an image from a cache (D) is far faster than performing a database read.

136. A, C. Route 53 offers a number of different routing policies: simple, failover, geolocation, geoproximity, latency-based, multivalue answer, and weighted.

137. A, B. Route 53 offers a number of different routing policies: simple, failover, geolocation, geoproximity, latency-based, multivalue answer, and weighted.

138. B, C. Route 53 offers a number of different routing policies: simple, failover, geolocation, geoproximity, latency-based, multivalue answer, and weighted.

139. C. Simple routing is ideal for sending all traffic to a single resource.

140. B. Failover routing is used to send traffic to a single resource but then to failover routing to a secondary resource if the first is unhealthy.

141. C. Geolocation routing uses the location of a user's DNS query to determine which route to use.

142. B. Latency-based routing uses the latency of regions to determine where routing should direct users.

143. C. Multivalue answer routing can direct requests to multiple resources and also performs health checks on those resources.

144. D. Weighted routing uses predefined weights to determine how traffic is routed across multiple resources.

145. A. When there is a single resource to which traffic should be directed, simple routing is the best option.

146. A, C. The two options here that are valid are geolocation and geoproximity routing, both of which consider the location of the user before routing that user to a resource. Geographical routing is not a valid routing policy for Route 53.

147. D. Weights are simply integers that can be summed to determine an overall weight and the fractional weights of each resource to which traffic is directed.

148. C. A weight of 0 removes the resource from service in a weighted routing policy.

149. A. In a weighted routing policy, the numerical weights are added up, and each resource's weight is divided by the sum of all the weights. In this case, the total weight is 400, so A is 25% of that (100/400), B is 25% (100/400), and C is 50% (200/400).

150. A, C. A simple routing policy allows single and multiple resources for both the primary and secondary resources, so A and C are true. Weighted policies do honor health checks (so B is false), and D is inaccurate as weight numbers do not affect health checks.

151. A, B. The issues here are geographical proximity from EU users and load on the database, which has high CPU utilization. Therefore, those problems must be addressed. ElastiCache (A) should reduce load on the RDS instance, and CloudFront (B) caches responses in a way that should serve EU users more quickly.

152. B, D. This is another memorization question. Valid instance types begin with T, M, C, R, X, Z, D, H, I, F, G, and P. Frankly, it's hard to memorize these; the questions like this aren't frequent, but they can sometimes appear. In this case, E and Q are not valid instance type prefixes.

153. B. IAM offers permissions for AWS resources as well as access rights for users of the AWS platform.

154. A, C. IAM controls permissions for resource-to-resource interaction as well as user access to the AWS console. It does not provide an authentication interface or single sign-on.

155. B. IAM stands for Identity and Access Management.

156. A, C. IAM only applies to permissions for users, roles, and groups and does not affect billing or cost or specific application feature accessibility.

157. B, C. IAM does handle user permissions for accessing AWS (A) and EC2-to-S3 access (D), so these are both true and therefore incorrect. It does not handle hosted application permissions (B) or relate to SNS, making B and C the correct answers.

158. B. IAM is not the managed service for handling MFA Delete setup on S3 buckets.

159. B. Anytime a single account in AWS is shared, you likely have a security risk.

160. C. The only requirement here is creating a sign-in link that is not the same as the root sign-in link. Turning on MFA for the root or all accounts is not required, and while it is common to create an IAM group at this stage, it is not required for access.

161. A, B. Users, groups, roles, permissions, and similar constructs are part of IAM. Organizations and organizational units are part of AWS Organizations, a different facility.

162. A, D. Users, groups, roles, permissions, and similar constructs are part of IAM.

163. A, C. In this case, you'd need to create a role that allows an EC2 instance to communicate with another AWS service, in this case S3. While a default role would probably cover this use case, you might also write a custom policy if you had particular needs for something other than the default role's allowances.

164. A, D. You can provide console access and programmatic access via IAM. Programmatic access incudes API and CLI access.

165. A, C. There are four types of policies in IAM: identity-based, resource-based, organization SCPs, and access control lists (ACLs).

166. B. IAM policies are written in JSON.

167. A, C. IAM policies can be attached to users, groups, and roles in the case of identity-based policies, and AWS services and components via resource-based policies.

168. B. MFA stands for Multi-Factor Authentication and can be enabled on a user account by IAM.

169. A, C. IAM aids in scalability primarily by consolidating and centralizing management of permissions, both to AWS users (A) and from instances to services (C).

170. C, D. IAM provides permissions, groups, users, and roles, and AWS Organizations provides logical groupings and account management. Both operate across all AWS resources.

171. C. Power user access is a predefined policy that allows access to all AWS services with the exception of group or user management within IAM.

172. D. Root users can perform all actions related to IAM.

173. A. Power users can work with managed services, but they cannot create (or otherwise manage) IAM users.

174. B. Although it might sound odd, AWS strongly recommends you delete your root user access keys and create IAM users for everyday use.

175. A, C. As a starting point, always consider that the root account is typically required for account-level operations, such as closing an account (A). It's also needed for very privileged access; in this case, that's creating a CloudFront key pair, which essentially provides signed access to applications and is a very trusted action. IAM does allow you to distribute user and policy management (B and D).

176. A, B. Affecting another account is generally something that requires root account level access. In this case, that's D, as well as restoration of user permissions (C). Both A and B are available to non-root users.

177. D. It is impossible to remove access for the AWS account's root user.

178. C. This is a bit of a "gimme" question but sometimes comes up. AWS firmly believes that root account access should be highly limited, but also not confined to a single user. C, having a very small group of engineers (ideally AWS certified) is the best approach to reducing root account level access as much as possible.

179. D. AWS defines and keeps updated a number of IAM policies for users, including Administrator, Billing, and Power User.

180. A, C. You will always need to provide non-root sign-in URLs for new users, so A is essential. The remaining answers are concerned with permissions, and of the choices (B, by the way, isn't an actual option), the Developer Power User policy is a much better fit than the View-Only User policy.

181. D. Unless your manager is both highly technical and working on actual development issues, D is the best option: It provides View-Only access to AWS without adding unneeded privileges for the manager.

182. D. The Data Scientist policy is designed for just this purpose: running queries used for data analytics and business intelligence

183. A. This is a System Administrator role. While Power User would give permissions to the same services, it is likely too permissive. Remember, the key with these questions is to find the role that allows the specified operations without going beyond those any more than is necessary.

184. B, C. It is impossible to remove a root user's access to EC2 instances (B). Further, IAM is concerned with the raw AWS resources, not access to running web applications (C).

185. D. IAM changes apply immediately to all users across the system; there is no lag, and no need to log out and back in (D).

186. A. New users have no access to AWS services. They are "bare" or "empty" or "naked" users, in that it is merely a login to the AWS console (if a URL is provided). They cannot make any changes to AWS services or even view services.

187. C, D. New users have no access to AWS services. They will need a URL to use for logging in (C) and permissions via a valid AWS group such as Administrators or power users. Options A and B refer to groups that are not predefined by AWS.

188. A, B. To access the console, users need a sign-in URL (A) and a username and password. This is not the access key ID and secret access key referenced in B. Therefore, A and B would effectively block a user from accessing the console. There is no Log In To Console box for users.

189. C. AWS usernames have to be unique across the AWS account in which that user exists.

190. B, D. Programmatic access requires an access key ID and a secret access key pair. Usernames and passwords are used for console access.

191. A, C. Console access requires a username and password. Access keys and pairs are used for programmatic access, not console access.

192. B. IAM policy documents are written in JSON.

193. A. Of these, SSO is single sign-on, IAM is more generally applied here, and JSON is the language used for policy documents. But SAML, the Security Assertion Markup Language, is used directly to implement single sign-on.

194. C. If you have an external Active Directory, you'd want to federate those users into AWS. This allows you to use the existing user base, not re-create each individual user.

195. D. A policy document is a collection of permissions in IAM.

196. D. IAM users are global to an AWS account and are not region-specific.

197. C. Like IAM users, policy documents are global. There are no changes or steps you need to take to make these work globally.

198. A, B. Auto Scaling is most focused on capacity management (B), ensuring that your applications can perform by keeping the capacity sufficient. Further, it performs a minimal amount of monitoring to effect this (A). It does not limit cost, although it does help in cost reduction, and it has nothing to do with permissions management.

199. A, C. Auto Scaling helps you to quickly set up scaling (C) and to then keep costs to a minimum (A). It does not affect network performance, and while there is a reduction of overhead, this is not related to maintaining individual VPCs (D).

200. A, C. Auto Scaling can be applied to both Aurora (and specifically read replicas) and DynamoDB.

201. A, D. EC2 instances as well as ECS containers can both be scaled up and down by Auto Scaling.

202. C. A collection of components, such as EC2 instances that will grow and shrink to handle load, is an Auto Scaling group.

203. B, D. When creating an Auto Scaling group, you can specify the minimum and maximum size as well as a desired capacity and scaling policy. You cannot specify how many instances to add at once, nor the desired cost.

204. A, B. When creating an Auto Scaling group, you can specify the minimum and maximum size as well as a desired capacity and scaling policy. While you can specify triggers that are used to grow or shrink the group, you can not specify a memory allocation or a minimum processing threshold (neither is an actual AWS term).

205. B, C. A launch configuration contains an AMI ID, key pair, instance type, security groups, and possibly a block device mapping.

206. B, C. A launch configuration contains an AMI ID, key pair, instance type, security groups, and possibly a block device mapping. Cluster size is not part of a launch configuration, although a maximum number of instances can be added to an Auto Scaling group. Maximum memory utilization also is not part of a launch configuration but can be a trigger for scaling.

207. A, C. There are a number of valid scaling policies for Auto Scaling: Maintain current instance levels, manual scaling, schedule-based scaling, and demand-based scaling.

208. A, D. There are a number of valid scaling policies for Auto Scaling: Maintain current instance levels, manual scaling, schedule-based scaling, and demand-based scaling. Resource-based scaling and instance-based scaling are not actual scaling policy options.

209. D. You can choose to maintain current instance levels at all times. This is essentially ensuring that no instances are added unless an instance fails its health checks and needs to be restarted or replaced.

210. A. Demand-based scaling allows you to specify parameters to control scaling. One of those parameters can be CPU utilization, so this is the policy you'd use for this use case.

211. B. This one should be pretty easy. Schedule-based scaling allows you to specify a particular time period during which resources should scale up or down.

212. C. Manual scaling allows you to specify a minimum and maximum number of instances as well as a desired capacity. The Auto Scaling policy then handles maintaining that capacity.

213. A. Manual scaling allows you to specify a minimum and maximum number of instances as well as a desired capacity. You would specify a time to scale up for a schedule-based policy and maximum CPU utilization as well as scaling conditions for a demand-based policy.

214. A, C. The most common approach is to use CloudWatch triggers—such as memory or CPU utilization—to notify AWS to scale a group up or down. However, you can also manually scale up or down with the AWS console.

215. C. While you can remove the instance altogether (B), you'd eventually want to put it back in the group, meaning you're incurring extra work. The best approach is to put the instance into Standby mode. This allows the group to scale up if needed, and then you can troubleshoot the instance and then put it back into the InService state when complete.

216. C, D. InService and Standby are valid states for an instance, while Deleted and ReadyForService are not.

217. B. You have to create a launch configuration first, then an Auto Scaling group, and then you can verify your configuration and group.

218. B. A launch configuration needs a single AMI ID to use for all instances it launches.

219. D. Security groups work for launch configurations just as they do with instances: You may use as many as you like.

220. D. All of these are valid options for creating an Auto Scaling group.

221. B, C. All of these are acceptable options, but the best options are to use the existing EC2 instance as a basis for a new Auto Scaling group and to set up demand-based scaling. Anytime you have an existing instance that is working, you can simply start from there, rather than using a launch configuration and duplicating the setup. Demand-based scaling will respond to changing conditions better than having to manually scale up and down or to set a desired capacity (which is unknown based on the question).

222. A, B. This is a case of having a recurring performance issue, which points to using schedule-based scaling. Further, you know that access is centered around the US East regions. C might help the issue, but without knowing more about the application, it's not possible to tell if caching content would significantly improve performance.

223. C. Here, the determining factor is the requirement of instant access. S3 One Zone-IA will give you that access, at a lower cost than S3 standard and S3-IA. According to AWS, all three classes have the same first byte latency (milliseconds).

224. D. Glacier takes 3–5 hours to deliver the first byte.

225. D. This is easy to miss, and often is. All three of these S3 storage classes share the same first-byte latency: milliseconds.

226. D. Spot instances offer you significant costs savings as long as you have flexibility and application processes can be stopped and started.

Domain 3: Specify Secure Applications and Architectures

1. B, D. Option A is false, but option B is true. Default security groups prevent all traffic in and allow all traffic out. Options C and D are about whether or not a security group is stateful: whether an incoming connection automatically can get back out. Security groups *are* stateful, so D is true. If the subject of the question was a NACL, then option C would be true, as NACLs are stateless.

2. B. D is not a good answer because relying on encryption outside of S3 does not best address the concerns around consistency. It is generally better to allow AWS to handle encryption in cases where you want to ensure all encryption is the same across a data store. SSE-C, SSE-KMS, and SSE-C all provide this. However, among those three, KMS is the best option for providing clear audit trails.

3. A, C. A bastion host is a publicly accessible host that allows traffic to connect to it. Then, an additional connection is made from the bastion host into a private subnet and the hosts within that subnet. Because the bastion must be accessed by public clients, it must be exposed to the Internet (A). If it is within a private subnet (B), it will not be accessible, making that answer incorrect. There also must be an explicit route from the bastion host into the private subnet (C); this is usually within a NACL. Finally, the security of the bastion *must* be different from the hosts in the private subnet. The bastion host should be hardened significantly as it is public, but also accessible; this is in many ways the *opposite* of the security requirements of hosts within a private subnet.

4. A, C. AWS sometimes asks questions like this to ensure that you understand that the root account is truly a root account and you cannot restrict that account's access. Anything that involves removing access for the root account is always invalid.

5. B. This is a "gimme question" that AWS will often ask on exams. You should never store your application keys on an instance, in an AMI, or anywhere else permanent on the cloud—meaning option B is true. Additionally, D makes no sense; application keys are for programmatic access, not console access.