

Hall Ticket Number:

--	--	--	--	--	--	--	--	--	--

I/IV B.Tech (Regular/Supplementary) DEGREE EXAMINATION

January , 2025

INFORMATION TECHNOLOGY

Seventh Semester

Cloud Programming

Time: Three Hours

Maximum: 70 Marks

Answer question 1 compulsorily.

(14X1 = 14 Marks)

Answer one question from each unit.

(4X14 = 56 Marks)

			CO	BL	M
1	a)	List any three features of Cloud	CO1	L1	1M
	b)	What is Virtualization	CO1	L1	1M
	c)	What is 5-4-3 principle of cloud computing	CO1	L1	1M
	d)	What is Amazon Cloud Watch?	CO1	L1	1M
	e)	What is EC2 AMI ID?	CO2	L1	1M
	f)	Provide the syntax to Create an AWSCredentials object?	CO2	L1	1M
	g)	List the advantages of SQS FIFO Queue.	CO2	L1	1M
	h)	What is AWS Lambda?	CO2	L1	1M
	i)	List few use cases of Amazon ECR	CO3	L1	1M
	j)	What is S3 Bucket.	CO3	L1	1M
	k)	Specify the CLI command syntax to rename an S3 object?	CO3	L1	1M
	l)	What is AWS Database Migration Service.	CO4	L1	1M
	m)	What is Auto Scaling?	CO4	L1	1M
	n)	Name some popular NoSQL databases offered by AWS.	CO4	L1	1M
Unit-I					
2	a)	Discuss about cloud architecture with neat diagram.	CO1	L3	7M
	b)	Explain in detail about cloud Service models with a neat diagram	CO1	L2	7M
(OR)					
3	a)	Explain in detail the phases and approaches of migrating an application to the Cloud.	CO1	L2	7M
	b)	Explain the anatomy of a typical Cloud environment.	CO1	L2	7M
Unit-II					
4	a)	Explain the key features of Amazon EC2 and different EC2 Instance Types.	CO2	L2	7M
	b)	Discuss managing EC2 instance using AWS SDK	CO2	L3	7M
(OR)					
5	a)	What is SQS? Explain managing SQS using management console	CO2	L2	7M
	b)	Explain managing SQS using AWS CLI.	CO2	L2	7M
Unit-III					
6		Explain in detail about Amazon EBS, Amazon ECR, and Amazon ECS	CO3	L2	14M
(OR)					
7	a)	Explain features of AWS S3	CO3	L2	7M
	b)	Explain the process of create, delete, empty bucket of Amazon S3 using AWS CLI	CO3	L2	7M
Unit-IV					
8		Explain in detailed about AWS RDS.	CO4	L2	14M
(OR)					
9	a)	Define NoSQL. Explain the various Database models and NoSQL Databases services supported by AWS.	CO4	L2	10M
	b)	Differentiate between SQL and NoSQL.	CO4	L3	4M



Schema Of Valuation

1. List any three features of Cloud:

- **On-Demand Self-Service:** Cloud computing allows users to provision computing resources like storage, networking, and processing power as needed, without needing manual intervention.
- **Scalability and Elasticity:** Cloud platforms provide the ability to scale resources up or down according to demand, enabling flexible use of resources.
- **Pay-as-You-Go:** Cloud services typically follow a pay-per-use model, where users pay only for the resources they use, making it cost-efficient.

2. What is Virtualization?

- Virtualization allows physical IT resources to provide multiple virtual images of themselves so that their underlying processing capabilities can be shared by multiple users.

3. What is the 5-4-3 Principle of Cloud Computing?

- 5 Essential Characteristics: → On-demand self-service, Broad network access, Elastic resource pooling, Rapid elasticity, Measured service
- 4 Cloud Deployment Models → Private, Public, Community, Hybrid
- 3 Service Offering Models → SaaS, PaaS, IaaS

4. What is Amazon CloudWatch?

- **Amazon CloudWatch** is a monitoring and observability service that provides real-time visibility into cloud resources and applications. It collects metrics, logs, and events, and allows users to set alarms, visualize performance, and automate actions based on the data.

5. What is EC2 AMI ID?

- **EC2 AMI ID** (Amazon Machine Image ID) is a unique identifier assigned to an Amazon Machine Image (AMI), which is a pre-configured template for launching EC2 instances. It contains the operating system, application server, applications, and configurations necessary to run an instance.

6. Provide the syntax to Create an AWSCredentials object:

```
AWSCredentials credentials = new BasicAWSCredentials("accessKey", "secretKey");
```

7. List the advantages of SQS FIFO Queue:

- **Message Ordering:** FIFO (First-In-First-Out) queues preserve the order of messages, which is important for tasks that depend on sequence.
- **Exactly-Once Processing:** FIFO queues ensure that each message is processed exactly once, preventing duplicates.

8. What is AWS Lambda?

- **AWS Lambda** is a serverless compute service that lets you run code in response to events without provisioning or managing servers. It automatically scales, handles availability, and is event-driven, meaning it can be triggered by various AWS services like S3, DynamoDB, or HTTP requests via API Gateway.

9. List a few use cases of Amazon ECR (Elastic Container Registry):

- **Storing Docker Images:** ECR is a fully managed container image registry that stores and manages Docker container images.
- **CI/CD Pipeline Integration:** ECR integrates with Amazon ECS and EKS for continuous integration and continuous deployment workflows.

- **Secure Container Storage:** ECR supports encryption at rest and access control policies to ensure secure storage of container images.

10. What is S3 Bucket?

- An **S3 Bucket** is a container in Amazon Simple Storage Service (S3) that stores objects (files). Buckets are the basic building blocks of S3, used to store data like documents, images, backups, and more.

11. Specify the CLI command syntax to rename an S3 object:

To rename an object in an S3 bucket, you generally need to copy the object to a new location with the new name and delete the old object:

```
aws s3 cp s3://bucket-name/old-object-name s3://bucket-name/new-object-name
```

```
aws s3 rm s3://bucket-name/old-object-name
```

12. What is AWS Database Migration Service (DMS)?

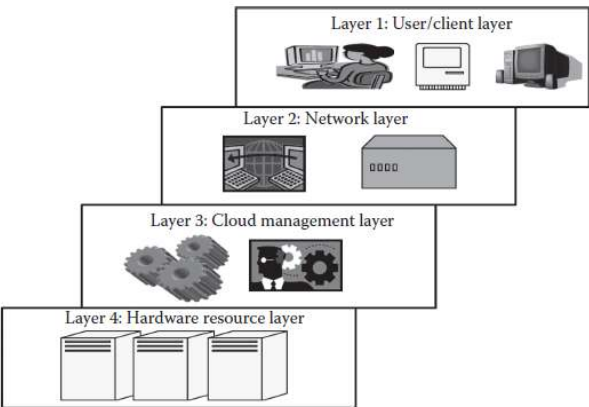
- **AWS Database Migration Service (DMS)** helps migrate databases to AWS quickly and securely. It supports both homogenous migrations (e.g., Oracle to Oracle) and heterogeneous migrations (e.g., Oracle to MySQL) while minimizing downtime during the migration process.

13. What is Auto Scaling?

- **Auto Scaling** automatically adjusts the number of EC2 instances in an application to maintain steady, predictable performance at the lowest possible cost. Auto Scaling ensures that you have the right amount of resources running based on demand.

14. Name some popular NoSQL databases offered by AWS.

- **Amazon DynamoDB:** A fully managed key-value and document database that provides fast and predictable performance.
- **Amazon DocumentDB:** A scalable document database service compatible with MongoDB.
- **Amazon Keyspaces (for Apache Cassandra):** A managed database service for running Cassandra workloads.
- **Amazon Neptune:** A graph database service suitable for building applications with connected data.

Unit-I		
2	a)	Discuss about cloud architecture with neat diagram.
7M		
<p>The cloud architecture can be divided into four layers based on the access of the cloud by the user. They are as follows.</p> <ol style="list-style-type: none"> 1. Layer 1 (User/Client Layer) 2. Layer 2 (Network Layer) 3. Layer 3 (Cloud Management Layer) 4. Layer 4 (Hardware Resource Layer) <p>The following diagram shows the Cloud Architecture.</p> <p style="text-align: center;">Diagram → 2M</p> <p>4 Layers Explanantion → 5M</p> <p>Layer 1 (User/Client Layer): This layer is the lowest layer in the cloud architecture. All the users or client belong to this layer. This is the place where the client/user initiates the connection to the cloud. The client can be any device such as a thin client, thick client, or mobile or any handheld device that would support basic functionalities to access a web application. The thin client here refers to a device that is completely dependent on some other system for its complete</p>		
		

functionality. thick clients are general computers that have adequate processing capability. Usually, a cloud application can be accessed in the same way as a web application. But internally, the properties of cloud applications are significantly different. Thus, this layer consists of client devices.

Layer 2 (Network Layer):

This layer allows the users to connect to the cloud. The whole cloud infrastructure is dependent on this connection where the services are offered to the customers. This is primarily the Internet in the case of a public cloud. The public cloud usually exists in a specific location and the user would not know the location as it is abstract. And, the public cloud can be accessed all over the world. In the case of a private cloud, the connectivity may be provided by a local area network (LAN). Even in this case, the cloud completely depends on the network that is used.

This layer does not come under the purview of service-level agreements (SLAs), that is, SLAs do not take into account the Internet connection between the user and cloud for quality of service (QoS).

Layer 3 (Cloud Management Layer):

This layer consists of software that are used in managing the cloud. The software can be a cloud operating system (OS), a software that acts as an interface between the data center (actual resources) and the user, or a management software that allows managing resources. These software usually allow resource management (scheduling, provisioning, etc.), optimization (server consolidation, storage workload consolidation), and internal cloud governance. This layer comes under the purview of SLAs, that is, the operations taking place in this layer would affect the SLAs that are being decided upon between the users and the service providers.

Layer 4 (Hardware Resource Layer):

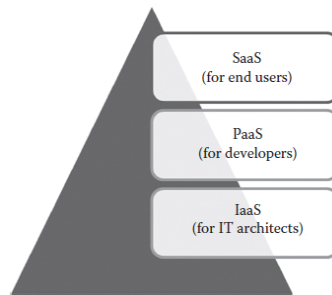
Layer 4 consists of provisions for actual hardware resources. Usually, in the case of a public cloud, a data center is used in the back end. Similarly, in a private cloud, it can be a data center, which is a huge collection of hardware resources interconnected to each other that is present in a specific location or a high configuration system. This layer comes under the purview of SLAs. This is the most important layer that governs the SLAs. This layer affects the SLAs most in the case of data centers. Whenever a user accesses the cloud, it should be available to the users as quickly as possible and should be within the time that is defined by the SLAs.

b) Explain in detail about cloud Service models with a neat diagram

7M

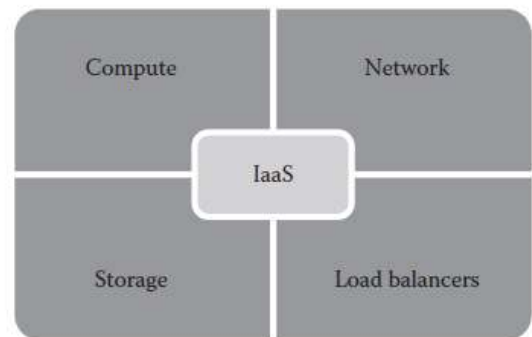
The three basic service models are: IaaS, PaaS, SaaS are shown below:

Diagram → 2M



3 Service Models → 5M

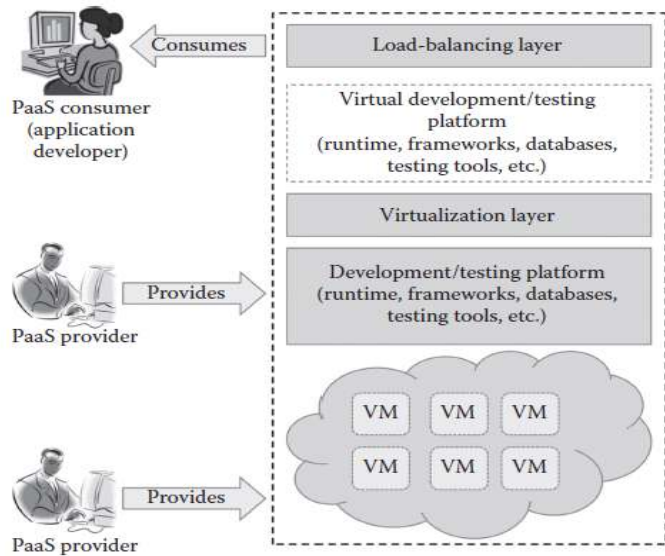
IaaS: The ability given to the infrastructure architects to deploy or run any software on the computing resources provided by the service provider. Here, the underlying infrastructures such as compute, network, and storage are managed by the service provider. Thus, the infrastructure architects are exempted from maintaining the data center or underlying infrastructure. The end users are responsible for managing applications that are running on top of the service provider cloud infrastructure.



✓ include Amazon Web Services (AWS), Google Compute Engine, OpenStack, and Eucalyptus. Generally, the IaaS services are provided from the service provider cloud data center. The end users can access the services from their devices through web command line interface (CLI) or application programming interfaces (APIs) provided by the service providers.

✓ Some of the popular IaaS providers include Amazon Web Services (AWS), Google Compute Engine, OpenStack, and Eucalyptus.

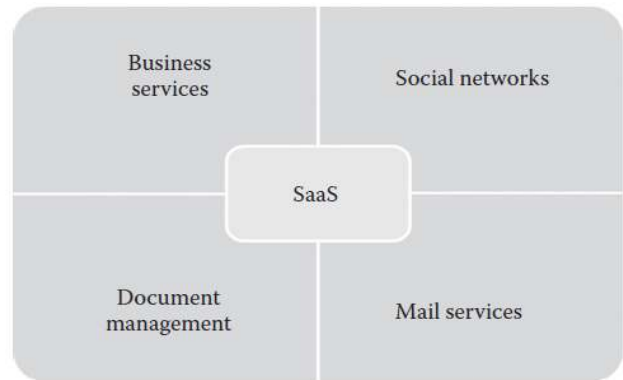
PaaS: The ability given to developers to develop and deploy an application on the development platform provided by the service provider. Thus, the developers are exempted from managing the development platform and underlying infrastructure. Here, the developers are responsible for managing the deployed application and configuring the development environment.



✓ Generally, PaaS services are provided by the service provider on an on-premise or dedicated or hosted cloud infrastructure. The developers can access the development platform over the Internet through web CLI, web user interface (UI), and integrated development environments (IDEs).

✓ Some of the popular PaaS providers include Google App Engine, Force.com, Red Hat OpenShift, Heroku, and Engine Yard.

SaaS: The ability given to the end users to access an application over the Internet that is hosted and managed by the service provider. Thus, the end users are exempted from managing or controlling an application, the development platform, and the underlying infrastructure.



✓ Generally, SaaS services are hosted in service provider–managed or service provider–hosted cloud infrastructure. The end users can access the services from any thin clients or web browsers.

✓ Some of the popular SaaS providers include Salesforce.com, Google Apps, and Microsoft office 365.

(OR)

3 a) Explain in detail the phases and approaches of migrating an application to the Cloud. 7M

- Cloud migration encompasses moving one or more enterprise applications and their IT environments from the traditional hosting type to the cloud environment, either public, private, or hybrid.
- Cloud migration presents an opportunity to significantly reduce costs incurred on applications. This activity comprises, of different phases like evaluation, migration strategy, prototyping, provisioning, and testing.

Phases of Cloud Migration:

1. *Evaluation*
2. *Migration strategy*
3. *Prototyping*
4. *Provisioning*
5. *Testing*

Evaluation: Evaluation is carried out for all the components like

- Current infrastructure and
- Application Architecture,
- Environment in terms of Compute, Storage, Monitoring, and Management,
- SLAs,
- Operational Processes,
- Financial Considerations,
- Risk,
- Security,
- Compliance, and
- Licensing needs

are identified to build a business case for moving to the cloud.

2. Migration strategy: Based on the evaluation, a migration strategy is drawn—a hot plug strategy is used where the applications and their data and interface dependencies are isolated and these applications can be operationalized all at once. Otherwise the application is partially migrated.

3. Prototyping:

Migration activity is preceded by a prototyping activity to validate and ensure that a **small portion of the applications are tested** on the cloud environment with test data setup.

4. Provisioning: Premigration optimizations identified are implemented.

- ✓ **Cloud servers** are provisioned for all the identified environments
- ✓ **Necessary platform software's** and applications are deployed
- ✓ **Configurations** are tuned to match the new environment sizing
- ✓ **Databases and files** are replicated
- ✓ **All internal and external integration points** are properly configured
- ✓ **Web services, batch jobs, and operation and management software** are set up in the new environments

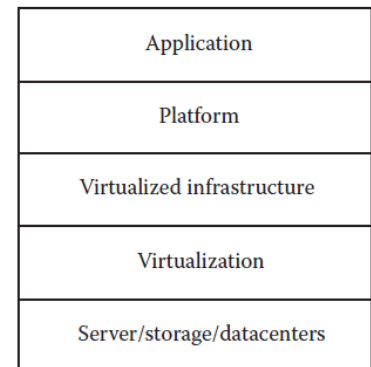
5. Testing: Postmigration tests are conducted to ensure that migration has been successful.

- ✓ Performance and Load testing
- ✓ Failure and Recovery testing
- ✓ Scale-out testing are conducted against the expected traffic load and resource utilization levels.

b) Explain the anatomy of a typical Cloud environment.

7M

- Cloud anatomy can be simply defined as the structure of the cloud. Cloud anatomy cannot be considered the same as cloud architecture.
- It may not include any dependency on which or over which the technology works, **whereas architecture wholly defines and describes the technology over which it is working.**
- Architecture is a hierarchical structural view that defines the technology as well as the technology over which it is dependent or/and the technology that are dependent on it. **anatomy can be considered as a part of architecture.**
- It depends on the person to choose the depth of description of the cloud. A different view of anatomy is given



There are basically **five components of the cloud:**

1. *Application:* The upper layer is the application layer. In this layer, any applications are executed.
2. *Platform:* This component consists of platforms that are responsible for the execution of the application. This platform is between the infrastructure and the application.
3. *Infrastructure:* The infrastructure consists of resources over which the other components work. This provides computational capability to the user.
4. *Virtualization:* Virtualization is the process of making logical components of resources over the existing physical resources. The logical components are isolated and independent, which form the infrastructure.
5. *Physical hardware:* The physical hardware is provided by server and storage units.

Unit-II

4 a) Explain the key features of Amazon EC2 and different EC2 Instance Types.

7M

- ✓ **Amazon Elastic Compute Cloud (EC2)** provides virtual computing. EC2 can be used for various purposes such as running custom applications, storing files, and so on.
- ✓ Users have a wide variety of operating systems to launch EC2 instances.
- ✓ Amazon EC2 is an elastic virtual server that resides under the AWS cloud environment.
- ✓ Users can deploy their software on the EC2 instance.
- ✓ Amazon has given control to the users to create, start, stop, and terminate the instance at their convenience.
- ✓ Amazon has a variety of Amazon Machine Images (AMIs) available that can be used to create new servers in the cloud. Users can even create their own AMIs from their EC2 instance.

EC2 Features:

- **Large number of OSs support:** Amazon EC2 provides a wide variety of operating systems that can be used to boot the server. Amazon Linux, Debian, SUSE, FreeBSD, CentOS, Red Hat Enterprise Linux, SUSE Linux Enterprise Server, Ubuntu, Windows, and other Linux operating systems.
- **Elasticity:** Amazon EC2 infrastructure is elastic in nature. Elasticity means the system manages its resources by provisioning and de-provisioning them without manual intervention and, therefore, manages load dynamically. You can configure the threshold limit when a new EC2 instance needs to be created based on the load on EC2 and terminate EC2 when the load on EC2 is below the threshold limit.

- **Fault tolerance and latency:** Amazon EC2 provides flexibility to auto-scale the servers on an as-needed basis. If one instance fails, another instance can be added to serve the purpose. Amazon also provides different availability zones where system architecture is designed in such a way that if a zone in a (region) is down, another zone in the same (region) can serve the request.
- **Pricing:** Amazon charges only for the hours consumed. There may be different pricing options available for different capacity instances. Along with EC2, if we also opt for storage, pricing is based on storage type and the amount of data that is being stored in it.
- **Security:** The security provided by EC2 is similar to the traditional firewall. It allows users to manage the accessibility of AWS resources to provide client information such as IP addresses or subnet groups. It is an effective way to manage security on EC2.
- **Service Commitment:** AWS provides the service commitment for its uptime of 99.95%. This means EC2 instances will be up and running for 99.95% of your monthly billing cycle. If this service commitment is not met, users will receive the service credit based on the criteria defined by AWS.

b)	Discuss managing EC2 instance using AWS SDK			7M
----	---	--	--	----

To connect with AWS for Java using SDK, you need to provide the AWSCredentials.

There are various ways to provide AWS credentials:

- ✓ Default Credential Provider
- ✓ Specify Credential Provider or Provider Chain
- ✓ Provide Access Key, and Secret Key

1. Default Credential Provider:

- ✓ This is the default provider when we don't provide any parameters while creating an AWS service client. DefaultAWSCredentialsProviderChain is the class that will be used to create the AWS credentials.
- ✓ This class will find the credentials in the following order:
 Environment Variables: This technique uses the EnvironmentVariableCredentialsProvider class that fetches values for AWS_ACCESS_KEY_ID and AWS_SECRET_ACCESS_KEY.
 System Properties: This technique is used to find the values for aws.accessKeyId and aws.secretKey in Java system properties.
- ✓ **Credentials Property File:** A default credentials property file can be loaded to get these credentials. This property file resides under C:\Users\{USER}\.aws\credentials or ~/.aws/credentials. This path differs based on the operating system
- Use **ProfileCredentialsProvider** to get the credentials details from the credentials folder in c:\users\{Username}\.aws\credentials
Ex:

```
AWSCredentials credentials = new ProfileCredentialsProvider("default").getCredentials();
```

2. Specify Credential Provider or Provider Chain:

There are different providers you can use to create the AWSCredentials object.

You can even create your own class by doing the following:

- Implementing AWSCredentialsProvider interface
- Sub-classing the AWSCredentialsProviderChain class

3. Provide Access Key and Secret Key Explicitly:

There may be cases when we need to explicitly pass the access key ID and secret access key to create an AWSCredentials object.

To create an AWSCredentials object, we will use the following:

```
AWSCredentials credentials = new BasicAWSCredentials("access_key_id", "secret_access_key");
```

Specifying default AWS region:

- You should set a default AWS Region that will be used for accessing AWS services with the AWS SDK for Java. For the best network performance, choose a region that's geographically close to you (or to your customers).
- If you don't select a region, then **us-east-1** will be used by default.

Specify in C:\Users\USERNAME\.aws\config file on Windows

```
[default]
region = ap-south-1
```

(OR)

5	a)	What is SQS? Explain managing SQS using management console			7M
---	----	--	--	--	----

SQS definition → 2M

SQS (Simple Queue Service): Amazon Simple Queue Service (SQS) is a fully managed message queuing service that enables you to decouple and scale microservices, distributed systems, and serverless applications.

AWS Management Console can be used to manage SQS. It provides an easy way to configure SQS. To configure SQS, log in to AWS Console. Under the All Services category, click SQS.

Any 5 operations → 5M

Operations that can be performed with the SQS using Management Console include:

- ✓ Create Queue
- ✓ Queue Listing
- ✓ Configure Queue Permission
- ✓ Send Message
- ✓ Receive Message
- ✓ Delete Message
- ✓ Purge Queue
- ✓ Delete Queue

Create Queue: Steps:

1. Open the Amazon SQS console at <https://console.aws.amazon.com/sqs/>.
2. Choose Create queue.
3. On the Create queue page, specify the correct region.
4. The Standard queue type is selected by default. Choose FIFO.
You can't change the queue type after you create a queue.
5. Enter a Name for your queue. The name of a FIFO queue must end with the .fifo suffix.
6. To create your queue with the default parameters, scroll to the bottom and choose Create Queue.
Amazon SQS creates the queue and displays the queue's Details page.

Configure Queue Permission: When you create a queue, by default only the queue owner can have access to the queue. If you need to provide queue access to another user, you also need to provide the necessary permission.

Steps:

1. Select the queue from the listing and click the Permissions tab
2. This will show the permissions provided to queue. To add permission, click the Add Permission button.
3. You can provide AWS account IDs of the users you want to allow or deny accessing the queue. Once you have finished providing AWS account IDs and selected necessary actions. Click the Add Permission button. This will create the permission.
4. If you want to edit the permission and edit using policy document, click the Edit Policy Document (Advanced) button.

Send Message: Steps:

1. To send a message, choose the queue, right-click, and select Send a Message from menu. This will open the Send a Message screen.
2. We added the message body "Hello World!!!" and clicked the Send Message button.
3. The message is successfully sent and you can see the message identifier and MD5 of the body in the success screen.
4. On closing the success screen, you can check on the queue listing that message is sent to and see that the "Messages Available" count is shown as 1. Once the message gets consumed, this count will decrease.

Receive Message: Receive Message will retrieve messages from the queue. You can provide the maximum number of messages to retrieve from the queue and the polling time in seconds to determine if any messages are available.

Steps:

1. To receive messages, select the queue, right-click it, and click View/Delete Messages from the menu.
2. Configure the parameters and click Start Polling for Messages. This action will retrieve messages from the queue. You can explicitly stop the polling or wait for the polling to complete.
3. You can see the two messages have been retrieved. These are the messages we sent in the Send Message section. To view the message attributes, click the More Details link on each message listed.
4. Find the message body that we have sent and click the Message Attributes tab to see the message attributes that we sent.

Delete Message: Delete Message will remove messages from the queue. SQS doesn't remove messages automatically when you retrieve messages. You explicitly need to delete messages.

Steps:

1. When you receive messages, select the check boxes of messages to delete and click the Delete Messages button. This will prompt for confirmation to delete the messages.
2. Click the Yes, Delete Checked Messages button.

Purge Queue: Purge Queue will delete all messages that are available in the queue. Once you purge the queue, deleted messages can't be retrieved. The deletion process takes nearly 60 seconds.

Steps:

1. To purge the queue, select the queue, right-click it, and click “Purge Queue” from the menu. This will prompt for confirmation.

Delete Queue: Delete Queue will delete the queue regardless if the queue is empty or not. The deletion process takes nearly 60 seconds.

Steps:

1. Select the queue, right-click the queue you want to delete, and click Delete Queue from the menu.
2. Click the Yes, Delete Queue button, and it will delete the queue.

b) Explain managing SQS using AWS CLI.

7M

Managing EC2 using AWS CLI involves handling the following: Security Group, Key Pair, EC2 instance

- **Security Groups:** Operations involved are: Creating, Listing, Deleting Security Groups, Attach an Inbound Rule to Security Group.
- **Key Pairs:** Create, List, Delete & Importing Key pairs
- **EC2 instance:** Describe Images, Select an Instance Type, Launch EC2, List EC2 Instances, Get Instance Status, Assign Instance Tags and Start-Stop-Terminate Instance.

Security Groups: Operations on Security Groups include Creating, Listing, Deleting Security Group:

To create a security group, execute the following command:

```
aws ec2 create-security-group --group-name ec2group --description "EC2 creation"
```

To list Security Group:

```
aws ec2 describe-security-groups --group-names EC2CreationViaAWSCLI
```

To Delete Security Group:

```
aws ec2 delete-security-group --group-name EC2grp
```

Attach an Inbound Rule to Security Group:

```
aws ec2 authorize-security-group-ingress --group-name EC2grp --protocol tcp --port 22 --cidr 0.0.0.0/0
```

Key Pairs: Create, List, Delete & Importing Key pairs

Creation: `aws ec2 create-key-pair --key-name EC2keyname`

Listing: `aws ec2 describe-key-pairs --key-name EC2keyname`

Delete: `aws ec2 delete-key-pair --key-name EC2keyname`

Import: `aws ec2 import-key-pair --key-name EC2keyname`

EC2 instance: Operations involved are describe images, select instance types, launch EC2, list EC2 instances, get instance status, create instance tags, and start-stop-terminate instances.

Describe Images: To get the AMI ID, we will describe images. This will return the public images and private images that are available.

```
aws ec2 describe-images --image-ids ami-775e4f16
```

Select an Instance Type: To select an instance type, you need to find out which instance type suits your application requirements.

Launch EC2: The following command launch a new EC2 instance that has one instance count, “EC2CreationViaAWSCLI” key pair, and “EC2CreationViaAWSCLI” security group.

```
aws ec2 run-instances --image-id ami-775e4f16 --instance-type t2.micro --count 1 --key-name EC2CreationViaAWSCLI --security-groups EC2CreationViaAWSCLI
```

List EC2 Instances:

To get the EC2 instance details based on the instanceId, use describe-instances:

```
aws ec2 describe-instances --instance-ids i-04275d1e63998a3d3
```

To get all EC2 instances details, use describe-instances without providing any filters:

```
aws ec2 describe-instances
```

Get Instance Status:

To get the instance status that we created, execute the following command:

```
aws ec2 describe-instance-status --instance-id i-0f6ed7e30cf57f488
```

Assign Instance Tags:

To assign tags to EC2 instances, execute the following command:

```
aws ec2 create-tags --resources i-0f6ed7e30cf57f488 --tags Key=Chapter,Value=2
```

Key=Environment,Value=Production

Start-Stop-Terminate Instance:

You may require your instance to start-stop manually. First of all, stop your instance using the following command:

```
aws ec2 stop-instances --instance-ids i-0f6ed7e30cf57f488
```

To start the instance, execute the following command:

```
aws ec2 start-instances --instance-ids i-0f6ed7e30cf57f488
```

Amazon Web Services (AWS) offers a suite of storage and container services that cater to different aspects of modern cloud applications, including data storage, container management, and orchestration. Here's an overview of **Amazon EBS**, **Amazon ECR**, and **Amazon ECS**, and how they fit into cloud architectures:

1. Amazon Elastic Block Store (EBS)

Purpose: Amazon EBS provides persistent block-level storage volumes for use with Amazon EC2 instances. It is often used for storing data that requires frequent access and needs to persist even when the associated EC2 instance is stopped or terminated.

Key Features:

- **Durability & Availability:** EBS volumes are replicated within an Availability Zone (AZ) to protect against data loss due to hardware failure.
- **Scalability:** You can dynamically scale the storage size and performance of your volumes as per application needs.
- **Snapshots:** EBS allows you to take point-in-time snapshots of your volumes, which can be used for backups or to create new volumes.
- **Types of EBS Volumes:** Different volume types are available based on the workload:
 - **General Purpose (SSD):** Ideal for boot volumes and small-to-medium-sized databases.
 - **Provisioned IOPS (SSD):** High-performance storage for I/O-intensive applications.
 - **Throughput Optimized HDD:** Cost-effective storage for frequently accessed, throughput-heavy workloads.
 - **Cold HDD:** For infrequently accessed data.

Use Cases:

- **Databases:** Storing data for relational or NoSQL databases.
- **File Storage:** Persistent storage for file systems.
- **Boot Volumes:** Operating system volumes for EC2 instances.

2. Amazon Elastic Container Registry (ECR)

Purpose: Amazon ECR is a fully managed container registry service that allows you to store, manage, and deploy Docker container images. It is integrated with AWS services like ECS and EKS for container orchestration.

Key Features:

- **Fully Managed:** ECR handles all aspects of container registry management, including the setup, scaling, and security of the container images.
- **Security:** Supports encryption at rest using AWS Key Management Service (KMS) and provides fine-grained access control via AWS Identity and Access Management (IAM).
- **Image Scanning:** ECR can scan images for vulnerabilities before they are deployed.
- **Lifecycle Policies:** You can set up rules to automatically delete old images based on your retention preferences.
- **Multi-Region Replication:** ECR supports cross-region replication, enabling you to store and access container images in multiple AWS regions.

Use Cases:

- **Containerized Application Deployment:** Store and manage Docker images for applications running in ECS, EKS, or AWS Fargate.
- **DevOps/CI/CD Pipelines:** Use ECR as part of continuous integration/continuous deployment (CI/CD) workflows to manage container images.
- **Microservices:** Store microservice container images for applications that use container-based architectures.

3. Amazon Elastic Container Service (ECS)

Purpose: Amazon ECS is a fully managed container orchestration service that allows you to run and scale Docker containers on AWS. It abstracts away the complexities of managing container clusters, making it easier to deploy and manage containerized applications.

Key Features:

- **Cluster Management:** ECS automatically manages the underlying compute infrastructure (EC2 instances or AWS Fargate) to run containers in a scalable, secure, and cost-effective manner.
- **Integration with AWS Services:** ECS integrates with other AWS services like Amazon ECR (for container image storage), IAM (for permissions), CloudWatch (for monitoring), and more.
- **Task Definitions:** Define the configuration for running containers, including the CPU, memory, and environment variables.
- **Service Auto Scaling:** Automatically adjusts the number of running tasks based on demand, ensuring high availability and performance.
- **Fargate Support:** ECS supports AWS Fargate, a serverless compute engine for containers that abstracts away the need to manage EC2 instances, allowing you to run containers without provisioning or managing the underlying infrastructure.

Use Cases:

- **Microservices Architecture:** ECS is ideal for running and managing microservices applications by orchestrating multiple containers.
- **Batch Processing:** Use ECS to run containerized batch processing workloads.
- **Hybrid Cloud:** ECS can be used to manage containers across on-premises and AWS environments.

How They Work Together

These services are often used together in modern cloud architectures to build, deploy, and manage scalable containerized applications:

- **Amazon EBS** provides persistent storage for EC2 instances running containers, ensuring that data is not lost even if containers or instances are terminated.
- **Amazon ECR** stores and manages the Docker images that are used in ECS to create and deploy containerized applications.
- **Amazon ECS** orchestrates the deployment, scaling, and management of containers, and can use EBS volumes (via EC2 instances) to persist application data.

(OR)

7	a)	Explain features of AWS S3			7M
---	----	----------------------------	--	--	----

	b)	Explain the process of create, delete, empty bucket of Amazon S3 using AWS CLI			7M
--	----	--	--	--	----

AWS CLI to manage AWS SQS involves the following operations:

- | | | |
|------------------------|------------------------|-----------------------------|
| ✓ create-queue | ✓ get-queue-attributes | ✓ change-message-visibility |
| ✓ get-queue-url | ✓ list-queues | ✓ delete-message |
| ✓ add-permission | ✓ send-message | ✓ delete-message-batch |
| ✓ remove-permission | ✓ send-message-batch | ✓ purge-queue |
| ✓ set-queue-attributes | ✓ receive-message | ✓ delete-queue |

Any 6 operations → 7M

create-queue: To create a queue, you need to pass the queue name
`aws sqs create-queue --queue-name SQSDeadLetterQueue`

To create a queue with specific attributes, add `--attributes` to the command:

```
aws sqs create-queue --queue-name Chapter3SQSQueueName --attributes file:///C:/Users/Dell/create-queue-attributes.json
```

get-queue-url: Using the `get-queue-url` command, you will be able to get the URL of an existing queue:

```
aws sqs get-queue-url --queue-name SQSQueueName
```

add-permission: This command provides access to the queue. By default, the queue owner has all controls to a queue. If different users need to access the queue, permission must be provided to them.

```
aws sqs add-permission --queue-url https://us-west-2.queue.amazonaws.com/acctid/Chapter3SQSQueueName --label SQSPermission --aws-account-ids MY_ACCOUNT_ID --actions "SendMessage" "ReceiveMessage" "DeleteMessage" "ChangeMessageVisibility" "GetQueueAttributes" "GetQueueUrl"
```

remove-permission: This command will remove the permission from the queue based on the label provided in the request:

```
aws sqs remove-permission --queue-url https://us-west-2.queue.amazonaws.com/YOUR_AWS_ACCOUNT_ID/Chapter3SQSQueueName --label Chapter3SQSPermission
```

set-queue-attributes:

The `set-queue-attributes` command sets the queue attributes values. You can provide attributes either in key value form separated by a comma (,) or in a JSON format:

```
aws sqs set-queue-attributes --queue-url https://us-west-2.queue.amazonaws.com/YOUR_AWS_ACCOUNT_ID/Chapter3SQSQueueName --attributes DelaySeconds=0,MaximumMessageSize=262144,MessageRetentionPeriod=1209600
```

get-queue-attributes:

The `get-queue-attributes` command provides the queue attributes:

```
aws sqs get-queue-attributes --queue-url https://us-west-2.queue.amazonaws.com/YOUR_AWS_ACCOUNT_ID/Chapter3SQSQueueName --attribute-names All
```

list-queues: The `list-queues` command provides the list of all queues available:

```
aws sqs list-queues
```

send-message: This will send message to a queue

```
aws sqs send-message --queue-url https://us-west-2.queue.amazonaws.com/YOUR_AWS_ACCOUNT_ID/Chapter3SQSQueueName --message-body "Hello AWS SQS!!!"
```

send-message-batch: This command enables sending messages in batches. The maximum number of messages that can be sent is 10 messages per batch:

```
aws sqs send-message-batch --queue-url https://us-west-2.queue.amazonaws.com/YOUR_AWS_ACCOUNT_ID/Chapter3SQSQueueName --entries file://C:/Users/Dell/send-message-batch.json
```

receive-message: This command receives messages from a queue. The maximum number of messages that can be received per request is 10 and the maximum number of default messages is 1:

```
aws sqs receive-message --queue-url https://us-west-2.queue.amazonaws.com/YOUR_AWS_ACCOUNT_ID/Chapter3SQSQueueName --max-number-of-messages 10 --visibility-timeout 20 --wait-time-seconds 10
```

change-message-visibility: This command will change the message's visibility timeout for a specified number of seconds. The maximum timeout you can configure is 0 to 43,200 seconds (that is, 12 hours).

```
aws sqs change-message-visibility --queue-url https://us-west-2.queue.amazonaws.com /acctid/Chapter3SQSQueueName --receipt-handle AQEBIvyCtihBCvEBNdhd.....z YgGHQ4ytgoM7aDqXTdQjkkywA== --visibilitytimeout 12000
```

delete-message: This command deletes messages from the queue. You need to pass the receipt handle of the message received:

```
aws sqs delete-message --queue-url https://us-west-2.queue.amazonaws.com/ ACCTID/ Chapter3SQSQueueName --receipt-handle AQEBwhE5t0xyrwLaCBMQv.....gWPXyqsAFV+A==
```

delete-message-batch: The delete-message-batch command deletes the messages in a batch of 10 messages. Each deleted message will have a response in output:

```
aws sqs delete-message-batch --queue-url https://us-west-2.queue.amazonaws.com/YOUR_AWS_ACCOUNT_ID/Chapter3SQSQueueName --entries Id=082e80fe-a0e8-41f4-89cc-f54cbcf7b91c,ReceiptHandle=AQEBxXGsIlusZ.....C3bX6gSDQ==
```

purge-queue: This command deletes all the messages available in the queue. Once messages are deleted, they can't be recalled back. The delete process may take 60 seconds:

```
aws sqs purge-queue --queue-url https://us-west-2.queue.amazonaws.com/AWS_ACCOUNT_ID /Chapter3SQSQueueName
```

delete-queue: The delete-queue command deletes the queue whether the queue contains messages or not. You can't recall messages after they have been deleted. The delete process may take 60 seconds.

```
aws sqs delete-queue --queue-url https://us-west-2.queue.amazonaws.com/ ACCOUNT_ID /Chapter3SQSQueueName
```

Unit-IV

8	Explain in detailed about AWS RDS.		14M
---	------------------------------------	--	-----

Definition → 1M

Amazon RDS is a fully managed relational database service. Amazon Relational Database Service (Amazon RDS) is a web service that makes it easier to set up, operate, and scale a relational database in the cloud. It provides cost-efficient and resizable capacity while automating time-consuming administration tasks such as hardware provisioning, database setup, patching and backups. It frees you to focus on your applications so you can give them the fast performance, high availability, security and compatibility they need.

Amazon RDS is available on several database instance types - optimized for memory, performance or I/O and provides you with six familiar database engines to choose from, including Amazon Aurora, PostgreSQL, MySQL, MariaDB, Oracle Database, and SQL Server.

You can use the AWS Database Migration Service to easily migrate or replicate your existing databases to Amazon RDS.

Features:

→ 2M

Scalable– Amazon RDS permits to scale the social database by utilizing the AWS Management Console or RDS-explicit API. We can either increase or decrease your RDS prerequisites within minutes.

Inexpensive– Using Amazon RDS, we pay just for the features that we actually use. There is no forthcoming and long-term payment.

Secure– Amazon RDS gives unlimited authority over the system to get to their database and their related services.

Automatic backups– Amazon RDS backs up everything in the database including exchange logs up to most recent five minutes and furthermore oversees programmed backup timings.

Software patching– Automatically gets all the most recent patches for the database programming. We can likewise indicate when the product ought to be fixed utilizing DB Engine Version Management.

Benefits:

- ✓ Easy to administer
- ✓ Highly scalable
- ✓ Available and durable
- ✓ Fast
- ✓ Secure
- ✓ Inexpensive
- ✓ reliable

Amazon CloudWatch metrics for Amazon RDS:

→ 2M

CPU utilization, Storage, Memory, SWAP usage, DB Connections, I/O, Latency, Throughput, Replica Lag etc...

Amazon RDS Storage Types:

General Purpose SSD – General Purpose SSD volumes offer cost-effective storage that is ideal for a broad range of workloads. These volumes deliver single-digit millisecond latencies and the ability to burst to 3,000 IOPS for extended periods of time. Baseline performance for these volumes is determined by the volume's size.

Provisioned IOPS – Provisioned IOPS storage is designed to meet the needs of I/O-intensive workloads, particularly database workloads, that require low I/O latency and consistent I/O throughput.

Magnetic – Amazon RDS also supports magnetic storage for backward compatibility. We recommend that you use General Purpose SSD or Provisioned IOPS for any new storage needs. The maximum amount of storage allowed for DB instances on magnetic storage is less than that of the other storage types.

AWS RDS - MySQL DB

Explanation→9M

1. Crating an Amazon RDS DB instance (CreateDBInstance)

- using Console
- using AWS CLI
- using RDS API

2. Connecting to an Amazon RDS DB instance

- Finding the connection information for a MySQL DB instance
- Connecting from the MySQL client
- using MySQL client (Mysql workbench)
- using RDS API
- MySQL commandline tool (mysql shell)/ AWS CLI

3. Getting DescribeDBInstances information

- *using Console*
- *using AWS CLI*
- *using RDS API*

4. Importing and Exporting data

5. Managing and RDS DB Instance

- *Starting, Stopping, Rebooting, Modifying, Renaming, Deleting DB instances*

6. Working with RDS events

AWS RDS - Connecting to Db

- The connection information for a DB instance includes its **endpoint, port, and a valid database user, such as the master user.**
- For example, suppose that an endpoint value is mydb.123456789012.us-east-1.rds.amazonaws.com. In this case, the port value is 3306, and the database user is admin.
- Given this information, you specify the following values in a connection string:
 - For host or **host name or DNS name**, specify mydb.123456789012.us-east-1.rds.amazonaws.com.
 - For **port**, specify 3306.
 - For **user**, specify admin.
- To connect to a DB instance, use any client for a DB engine. For example, you might use the mysql utility to connect to a MariaDB or MySQL DB instance.

Getting Connection Information:

Console

To find the connection information for a DB instance in the AWS Management Console

1. Sign in to the AWS Management Console and open the Amazon RDS console at <https://console.aws.amazon.com/rds/>.
2. In the navigation pane, choose **Databases** to display a list of your DB instances.
3. Choose the name of the MySQL DB instance to display its details.
4. On the **Connectivity & security** tab, copy the endpoint. Also, note the port number. You need both the endpoint and the port number to connect to the DB instance.

Ex: From Windows command prompt, give command to get connection information:

```
aws rds describe-db-instances --filters "Name=engine,Values=mysql" --query
"*[].[DBInstanceIdentifier,Endpoint.Address,Endpoint.Port,MasterUsername]"
```

To connect to a DB instance using the MySQL client, enter the following command at a Windows command prompt to connect to a DB instance using the MySQL client.

- For the -h parameter, substitute the DNS name (endpoint) for your DB instance.
- For the -P parameter, substitute the port for your DB instance.
- For the -u parameter, substitute the user name of a valid database user, such as the master user. Enter the master user password when prompted.

Ex: To connect with mydb1id mysql database

```
C:\Users\bh>mysql -h mydb1id.c90djrnrzvuh.us-east-1.rds.amazonaws.com -P 3306 -u admin -p
```

Output:

```
Enter password: *****
```

```
Welcome to the MariaDB monitor. Commands end with ; or \g.
```

```
Your MySQL connection id is 193
```

```
Server version: 8.0.20 Source distribution
```

Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

MySQL [(none)]>

(OR)

9	a)	Define NoSQL. Explain the various Database models and NoSQL Databases services supported by AWS.			10M
---	----	--	--	--	-----

- NoSQL databases enable you to store data with flexible schema and a variety of data models. These databases are relatively easy for developers to use, and have the high performance and functionality needed for modern applications.
- NoSQL is a new breed of database management systems that fundamentally differ from relational database systems. NoSQL database is a highly scalable and flexible database management system. NoSQL database allows the user to store and process unstructured data and semi-structured data; this feature is not possible in RDBMS tools.
- NoSQL database is a type of non-relational database, and it is capable of processing structured, semi-structured and unstructured data.
- NoSQL databases, also referred to as “non-SQL” and “Not Only SQL” databases, are mainly used for unstructured data.
- **Data is not stored in tabular format** but is stored mainly **in documents, key-value pairs, graphs, or wide column stores format**. As NoSQL databases are schema agnostic, **unstructured data such as blog articles, photos, videos, or other content can be stored very easily**.

There are six types of NoSQL database models you can choose from in AWS.

- Key-Value Databases
- Document Databases
- Wide Column Databases
- Graph Databases
- Time Series Databases
- Ledger Databases

Key-Value Databases: Key-value databases enable you to store data in pairs containing a unique ID and a data value. This provides a flexible storage structure since values are not assigned to a table and can hold any amount or structure of data. These databases can manage large volumes of data or requests.

- ✓ Use cases for key-value databases include gaming applications, eCommerce systems, and high traffic applications.

Document Databases: Document databases are structured similarly to key-value databases except that keys and values are stored in documents written in a markup language like JSON, XML, or YAML. You can use these databases to store hierarchies of data by linking documents. Use cases for document databases include user profiles, catalogues, and content management.

- ✓ **AWS service:** Amazon DocumentDB, DynamoDB

Wide Column Databases: Wide column databases are based on tables but without a strict column format. Rows do not need a value in every column and segments of rows and columns containing different data formats can be combined. Use cases for wide column databases include route optimization, fleet management, and industrial maintenance applications.

- ✓ **AWS service:** Amazon Keyspaces (for Apache Cassandra)

Graph Databases: Graph databases are structured as collections of edges and nodes. Nodes are the individual data values and edges are the relationships between those values. These databases enable you to track intricately related data in an organic network rather than a structured table. Use cases for graph databases include recommendation engines, social networking, and fraud detection.

- ✓ **AWS service:** Amazon Neptune

Time Series Databases: Time series databases store data in time ordered streams. Data is not sorted by value or ID but by the time of collection, ingestion, or other timestamps included in the metadata.

These databases enable you to manage and query data based on time intervals. Use cases for time series databases include industrial telemetry, DevOps, and Internet of things (IoT) applications.

- ✓ **AWS service:** Amazon Timestream

Ledger Databases: Ledger databases are based on logs that record events related to data values. These logs are transparent, immutable, and can be verified cryptographically to prove the authenticity and integrity of data. Use cases for ledger databases include banking systems, registrations, supply chains, and systems of record.

- ✓ **AWS service:** Amazon Quantum Ledger Database (QLDB)

AWS service: Amazon DynamoDB

- AWS NoSQL Databases Services
 - ✓ Amazon DynamoDB
 - ✓ Amazon ElastiCache
 - ✓ Amazon Neptune
 - ✓ Amazon Timestream

- ✓ Amazon QLDB
- ✓ Amazon DocumentDB
- ✓ Amazon Keyspaces

Amazon DynamoDB : Amazon DynamoDB is a document and key-value database. It is a fully managed service that includes features for backup and restore, in-memory caching, security, and multiregion, multimaster distribution. DynamoDB supports atomicity, consistency, isolation, durability (ACID) transactions and encryption by default.

Amazon ElastiCache : Amazon ElastiCache is an in-memory data store that you can use in place of a disk-based database. It provides fully managed support for Memcached and Redis, and enables scaling with memory sharding. It is designed to support sub-millisecond response times and is typically used for queuing, real-time analytics, caching, and session stores.

Amazon Neptune : Amazon Neptune is a graph database service that is fully managed and optimized for storing data on billions of relationships. It supports a range of graph models and query languages, including W3C's RDF, Property Graph, SPARQL, and TinkerPop Gremlin.

Neptune includes features for point-in-time recovery, multi-zone data replication, continuous backups, and read replicas. It supports ACID transactions and provides encryption in-transit and at-rest.

Amazon Timestream : Amazon Timestream is a fully managed time series database with an adaptive query processing engine. It is a serverless service and automatically manages hardware and software maintenance and provisioning for you. Timestream includes features for automated data compression, tiering, retention, and rollups. It also includes built-in analytics for the approximation, smoothing, and interpolation of data.

Amazon QLDB : Amazon QLDB is a ledger database that you can use to track data changes. It is fully managed and designed to enable you to avoid complex setups required for managing ledger data with relational databases or blockchain.

QLDB provides a SQL-like API, full transactional support, and a flexible document data model. It includes features for automatic scaling, ACID compliant transactions, multizone availability, and data streaming with Kinesis Data Streams.

Amazon DocumentDB : Amazon DocumentDB is a fully managed document database that is compatible with MongoDB. DocumentDB architecture separates compute and storage resources for greater scalability and flexibility. It also includes support for up to 15 read replicas, data replication for durability across three availability zones, and free use of the AWS Database Migration Service.

Amazon Keyspaces : Amazon Keyspaces is a managed wide column database that is compatible with Apache Cassandra. You can use it to migrate Cassandra workloads and applications and continue to use Cassandra native code and tools. It includes features for autoscaling and enables you to select between on-demand or provisioned resources.

b) Differentiate between SQL and NoSQL.

4M

SQL	NoSQL
Relational Data Base Management System (RDBMS)	Non-relational or distributed database system.(Non-RDBMS)
These databases have fixed or static or predefined schema	They have dynamic schema
These databases are not suited for hierarchical data storage.	These databases are best suited for hierarchical data storage.
These databases are best suited for complex queries	These databases are not so good for complex queries
Vertically Scalable	Horizontally scalable
Follows ACID (Automocity, Consistency, Isolation and Durability) property	Follows CAP (consistency, availability, partition tolerance)