Storage - Terminologies

Decibel vs binary terminology

Networking				Storage			
Decimal Name	Decimal Abbr.	Decimal Power	Decimal Value	Binary Name	Binary Abbr.	Binary Power	Binary Value
Kilobyte	KB	10^3	1,000	Kibibyte	kiB	2^10	1,024
Megabyte	MB	10^6	1,000,000	Mebibyte	MiB	2^20	1,048,576
Gigabyte	GB	10^9	1,000,000,000	Gibibyte	GiB	2^30	1,073,741,824
Terabyte	TB	10^12	1,000,000,000,000	Tebibyte	TiB	2^40	1,099,511,627,776

AWS - SNOW family (storage t/f)

AWS Snowball

- AWS Snowball is a petabyte-scale data transport solution that uses secure appliances to transfer large amounts of data into and out of the AWS cloud.
- Using Snowball addresses common challenges with large-scale data transfers including high network costs, long transfer times, and security concerns.
- Transferring data with Snowball is simple, fast, secure, and can be as little as one-fifth the cost of high-speed Internet.

- Snowball is a strong choice for data transfer if you need to more securely and quickly transfer terabytes to many petabytes of data to AWS. Snowball can also be the right choice if you don’t want to make expensive upgrades to your network infrastructure, if you frequently experience large backlogs of data, if you're located in a physically

isolated environment, or if you're in an area where high-speed Internet connections are not available or cost-prohibitive.

- As a rule of thumb, if it takes more than one week to upload your data to AWS using the spare capacity of your existing Internet connection, then you should consider using Snowball. For example, if you have a 100 Mb connection that you can solely dedicate to transferring your data and need to transfer 100 TB of data, it takes more than 100 days to complete data transfer over that connection. You can make the same transfer by using multiple Snowballs in about a week.

Available Internet Connection	Theoretical Min. Number of Days to Transfer 100TB at 80% Network Utilization	When to Consider AWS Snowball?
T3 (44.736Mbps)	269 days	2TB or more
100Mbps	120 days	5TB or more
1000Mbps	12 days	60TB or more

Snowball Vs Snowball Edge

Snowball is suitable for the following use cases:

- Import data into Amazon S3
- Export from Amazon S3

Snowball Edge is suitable for the below:

- Import data into Amazon S3
- Export from Amazon S3
- Durable local storage
- Local compute with AWS Lambda
- Local compute instances
- Use in a cluster of devices
- Use with AWS Greengrass (IoT)
- Transfer files through NFS with a GUI

Take note that both services provide Import and Export data capabilities to and from Amazon S3. Notably, there are 6 other use cases where Snowball Edge is more suitable. Hence, the correct answers are the following options:

1. ***Local compute with AWS Lambda***

2. ***Local compute instances***

3. ***Use with AWS Greengrass (IoT)***

4. ***Transfer files through NFS with a GUI***

References:

https://docs.aws.amazon.com/snowball/latest/developer-guide/device-differences.html

https://aws.amazon.com/snowball-edge/

https://aws.amazon.com/snowball

Check out this AWS Snowball Cheat Sheet:

https://tutorialsdojo.com/aws-cheat-sheet-aws-snowball/

Check out this AWS Snowball Edge Cheat Sheet:

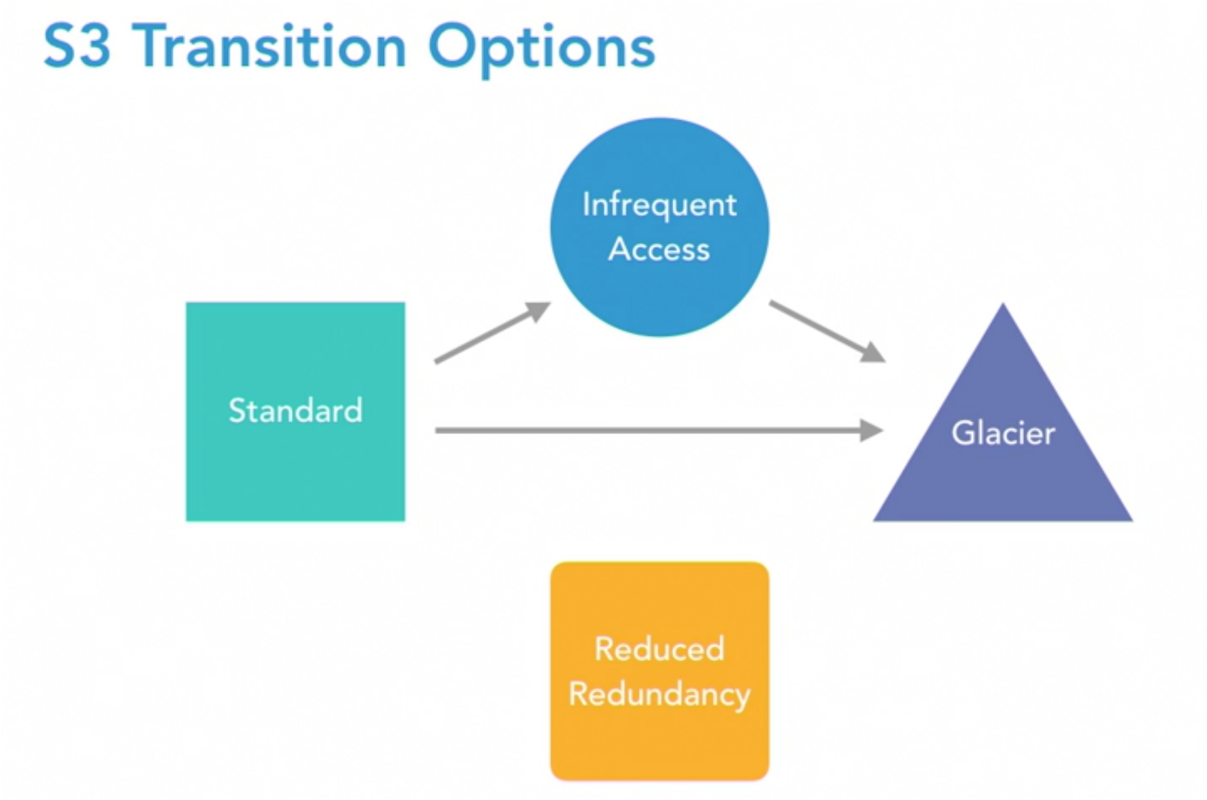
https://tutorialsdojo.com/aws-cheat-sheet-aws-snowball-edge/

S3 Transfer Acceleration vs Direct Connect vs VPN vs Snowball vs Snowmobile:

https://tutorialsdojo.com/aws-cheat-sheet-s3-transfer-acceleration-vs-direct-connect-vs-vpn-vs-snowball-vs-snowmobile/

S3

- Amazon S3 now provides increased performance to support at least 3,500 (PUTS) requests per second to add data and 5,500 (GETS) requests per second to retrieve data, which can save significant processing time for no additional charge.
- Each S3 prefix can support these request rates, making it simple to increase performance significantly.
- Applications running on Amazon S3 today will enjoy this performance improvement with no changes, and customers building new applications on S3 do not have to make any application customizations to achieve this performance. Amazon S3's support for parallel requests means you can scale your S3 performance by the factor of your compute cluster, without making any customizations to your application.
- Performance scales per prefix, so you can use as many prefixes as you need in parallel to achieve the required throughput. There are no limits to the number of prefixes.


S3 Transition Options
Infrequent Access
Standard
Glacier
Reduced Redundancy

	S3 Standard	S3 Intelligent-Tiering*	S3 Standard-IA	S3 One Zone-IA†	S3 Glacier	S3 Glacier Deep Archive
Designed for durability	99.999999999% (11 9's)	99.999999999% (11 9's)	99.999999999% (11 9's)	99.999999999% (11 9's)	99.999999999% (11 9's)	99.999999999% (11 9's)
Designed for availability	99.99%	99.9%	99.9%	99.5%	99.99%	99.99%
Availability SLA	99.9%	99%	99%	99%	99.9%	99.9%
Availability Zones	≥3	≥3	≥3	1	≥3	≥3
Minimum capacity charge per object	N/A	N/A	128KB	128KB	40KB	40KB
Minimum storage duration charge	N/A	30 days	30 days	30 days	90 days	180 days
Retrieval fee	N/A	N/A	per GB retrieved	per GB retrieved	per GB retrieved	per GB retrieved
First byte latency	milliseconds	milliseconds	milliseconds	milliseconds	select minutes or hours	select hours
Storage type	Object	Object	Object	Object	Object	Object
Lifecycle transitions	Yes	Yes	Yes	Yes	Yes	Yes

	Standard	**Standard - Infrequent Access**	**Reduced Redundancy Storage**
Durability	99.999999999%	99.999999999%	99.99%
Availability	99.99%	99.9%	99.99%
Concurrent facility fault tolerance	2	2	1
SSL support	Yes	Yes	Yes
First byte latency	Milliseconds	Milliseconds	Milliseconds
Lifecycle Management Policies	Yes	Yes	Yes

Reduced Redundancy Storage (RRS) is an Amazon S3 storage option that enables customers to store noncritical, reproducible data at lower levels of redundancy than Amazon S3's standard storage.

It provides a highly available solution for distributing or sharing content that is durably stored elsewhere, or for storing thumbnails, transcoded media, or other processed data that can be easily reproduced.

The RRS option stores objects on multiple devices across multiple facilities, providing 400 times the durability of a typical disk drive, but does not replicate objects as many times as standard Amazon S3 storage.

Reduced Redundancy Storage is:

- Backed with the Amazon S3 Service Level Agreement for availability.

- Designed to provide 99.99% durability and 99.99% availability of objects over a given year. This durability level corresponds to an average annual expected loss of 0.01% of objects.
- Designed to sustain the loss of data in a single facility.

AWS - S3 Encryption Techniques

Data protection refers to protecting data while in-transit (as it travels to and from Amazon S3) and at rest (while it is stored on disks in Amazon S3 data centers).

You can protect data in transit by using SSL or by using client-side encryption.

You have the following options for protecting data at rest in Amazon S3:

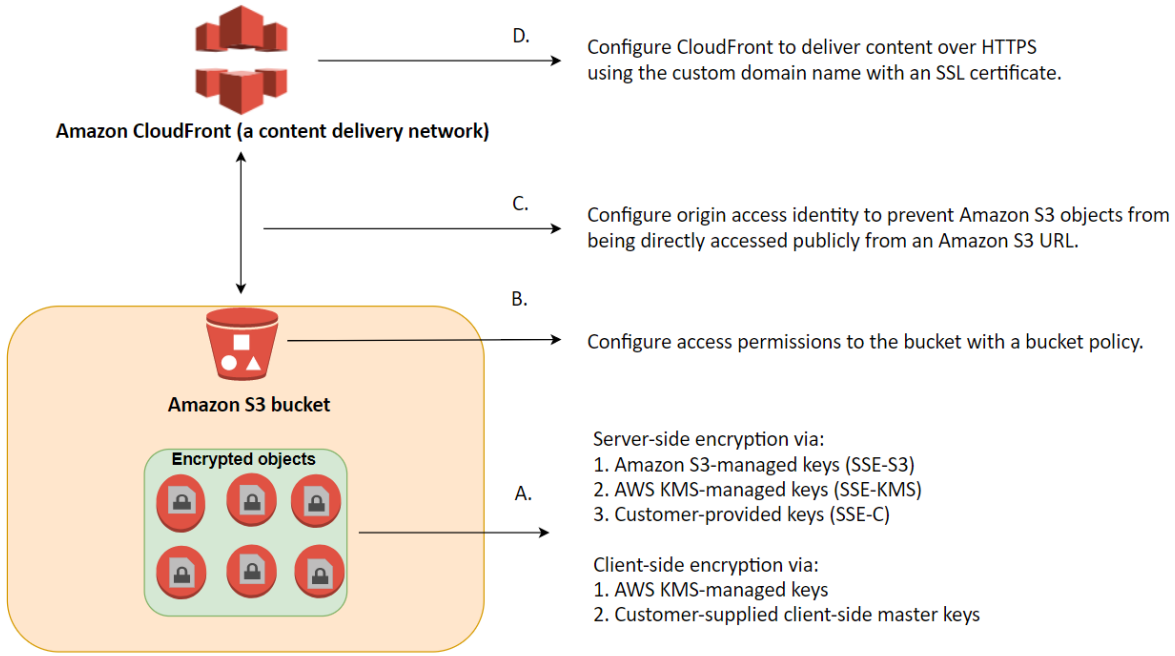
- Server-Side Encryption
- Client-Side Encryption

Use Server-Side Encryption – You request Amazon S3 to encrypt your object before saving it on disks in its data centers and decrypt it when you download the objects.

- Use Server-Side Encryption with Amazon S3-Managed Keys (SSE-S3)
- Use Server-Side Encryption with AWS KMS-Managed Keys (SSE-KMS)
- Use Server-Side Encryption with Customer-Provided Keys (SSE-C)

Use Client-Side Encryption – You can encrypt data client-side and upload the encrypted data to Amazon S3. In this case, you manage the encryption process, the encryption keys, and related tools.

- Use Client-Side Encryption with AWS KMS–Managed Customer Master Key (CMK)
- Use Client-Side Encryption Using a Client-Side Master Key



Reference:

http://docs.aws.amazon.com/AmazonS3/latest/dev/UsingEncryption.html

Check out this Amazon S3 Cheat Sheet:

https://tutorialsdojo.com/aws-cheat-sheet-amazon-s3/

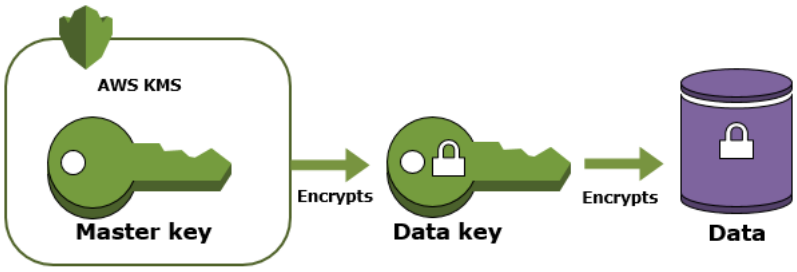
References:

https://docs.aws.amazon.com/AmazonS3/latest/dev/serv-side-encryption.html

https://docs.aws.amazon.com/AmazonS3/latest/dev/UsingClientSideEncryption.html

- Server-side encryption is the encryption of data at its destination by the application or service that receives it.
- AWS Key Management Service (AWS KMS) is a service that combines secure, highly available hardware and software to provide a key management system scaled for the cloud.
- Amazon S3 uses AWS KMS customer master keys (CMKs) to encrypt your Amazon S3 objects.
  - SSE-KMS encrypts only the object data.
  - Any object metadata is not encrypted.
- If you use customer-managed CMKs, you use AWS KMS via the AWS Management Console or AWS KMS APIs to centrally create encryption keys, define the policies that control how keys can be used, and audit key usage to prove that they are being used correctly.

- You can use these keys to protect your data in Amazon S3 buckets.



References:

https://docs.aws.amazon.com/AmazonS3/latest/dev/serv-side-encryption.html

https://docs.aws.amazon.com/AmazonS3/latest/dev/UsingKMSEncryption.html

https://docs.aws.amazon.com/kms/latest/developerguide/services-s3.html

Envelope encryption

- A customer master key (CMK) is a logical representation of a master key.
- The CMK includes metadata, such as the key ID, creation date, description, and key state.
- The CMK also contains the key material used to encrypt and decrypt data.
- You can use a CMK to encrypt and decrypt up to 4 KB (4096 bytes) of data.
- Typically, you use CMKs to generate, encrypt, and decrypt the data keys that you use outside of AWS KMS to encrypt your data. This strategy is known as **envelope encryption.**

You have three mutually exclusive options depending on how you choose to manage the encryption keys:

- **Use Server-Side Encryption with Amazon S3-Managed Keys (SSE-S3)** –
  - Each object is encrypted with a unique key.
  - As an additional safeguard, it encrypts the key itself with a master key that it regularly rotates.
  - Amazon S3 server-side encryption uses one of the strongest block ciphers available, 256-bit Advanced Encryption Standard (AES-256), to encrypt your data.
- **Use Server-Side Encryption with Customer Master Keys (CMKs) Stored in AWS Key Management Service (SSE-KMS)** –
  - Similar to SSE-S3, but with some additional benefits and charges for using this service.
  - There are separate permissions for the use of a CMK that provides added protection against unauthorized access of your objects in Amazon S3.

- SSE-KMS also provides you with an audit trail that shows when your CMK was used and by whom.
- Additionally, you can create and manage customer-managed CMKs or use AWS managed CMKs that are unique to you, your service, and your Region.

- **Use Server-Side Encryption with Customer-Provided Keys (SSE-C)** –
  - You (customer) manage the encryption keys and Amazon S3 manages the encryption, as it writes to disks, and decryption when you access your objects.

AWS - S3 Versioning

S3 versioning

- To control the versions of files that are served from your distribution, you can
  - either invalidate files
  - or give them versioned file names.

- If you want to update your files frequently, AWS recommends that you primarily use file versioning for the following reasons:
- - Versioning enables you to control which file a request returns even when the user has a version cached either locally or behind a corporate caching proxy. If you invalidate the file, the user might continue to see the old version until it expires from those caches.
- - CloudFront access logs include the names of your files, so versioning makes it easier to analyze the results of file changes.
- - Versioning provides a way to serve different versions of files to different users.
- - Versioning simplifies rolling forward and back between file revisions.
- - Versioning is less expensive. You still have to pay for CloudFront to transfer new versions of your files to edge locations, but you don't have to pay for invalidating files.

For Amazon S3 REST API calls, you have to include the following HTTP Request Headers:

x-amz-server-side-encryption-customer-algorithm

x-amz-server-side-encryption-customer-key

x-amz-server-side-encryption-customer-key-MD5

For presigned URLs, you should specify the algorithm using the** x-amz-server-side-encryption-customer-algorithm **request header.

References:

https://docs.aws.amazon.com/AmazonCloudFront/latest/DeveloperGuide/UpdatingExistingObjects.html

https://aws.amazon.com/premiumsupport/knowledge-center/prevent-cloudfront-from-caching-files/

https://docs.aws.amazon.com/AmazonCloudFront/latest/DeveloperGuide/Invalidation.html#PayingForInvalidation

Check out this Amazon CloudFront Cheat Sheet:

https://tutorialsdojo.com/aws-cheat-sheet-amazon-cloudfront/

AWS - S3 Event Notifications

References:

https://docs.aws.amazon.com/AmazonS3/latest/dev/NotificationHowTo.html

https://docs.aws.amazon.com/AmazonS3/latest/dev/ways-to-add-notification-config-to-bucket.html

https://aws.amazon.com/blogs/aws/s3-event-notification/

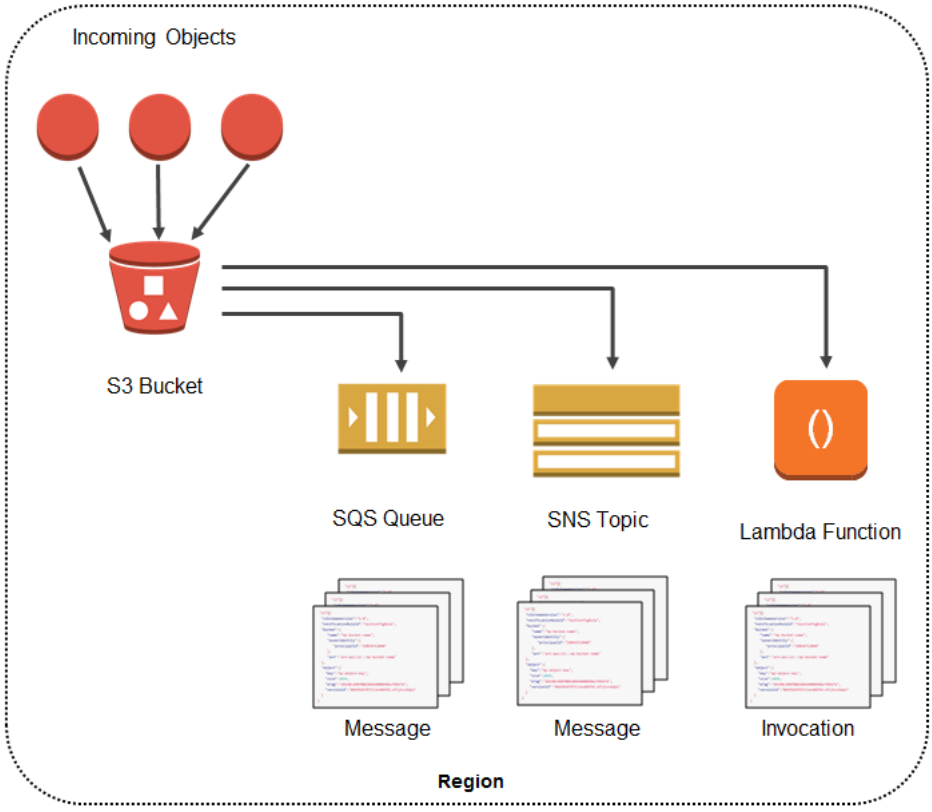
Check out this Amazon S3 Cheat Sheet:

https://tutorialsdojo.com/aws-cheat-sheet-amazon-s3/

Event Notifications

- The Amazon S3 notification feature enables you to receive notifications when certain events happen in your bucket.
- To enable notifications, you must first add a notification configuration that identifies the events you want Amazon S3 to publish and the destinations where you want Amazon S3 to send the notifications.
- You store this configuration in the *notification* sub-resource that is associated with a bucket. Amazon S3 provides an API for you to manage this sub-resource.

- Amazon S3 event notifications typically deliver events in seconds but can sometimes take a minute or longer.
- If two writes are made to a single non-versioned object at the same time, it is possible that only a single event notification will be sent.
- If you want to ensure that an event notification is sent for every successful write, you can enable versioning on your bucket.
- With versioning, every successful write will create a new version of your object and will also send an event notification.



Amazon S3 can publish notifications for the following events:

1. New object created events
2. Object removal events
3. Restore object events
4. Reduced Redundancy Storage (RRS) object lost events
5. Replication events

Event types	Description
*s3:ObjectCreated:** *s3:ObjectCreated:Put* *s3:ObjectCreated:Post* *s3:ObjectCreated:Copy* *s3:ObjectCreated:CompleteMultipartUpload*	Amazon S3 APIs such as PUT, POST, and COPY can create an object. Using these event types, you can enable notification when an object is created using a specific API, or you can use the *s3:ObjectCreated:** event type to request notification regardless of the API that was used to create an object. You do not receive event notifications from failed operations.
*s3:ObjectRemoved:** *s3:ObjectRemoved:Delete* *s3:ObjectRemoved:DeleteMarkerCreated*	By using the *ObjectRemoved* event types, you can enable notification when an object or a batch of objects is removed from a bucket. You can request notification when an object is deleted or a versioned object is permanently deleted by using the *s3:ObjectRemoved:Delete* event type. Or you can request notification when a delete marker is created for a versioned object by using *s3:ObjectRemoved:DeleteMarkerCreated*. You can also use a wildcard `s3:ObjectRemoved:*` to request notification anytime an object is deleted. You do not receive event notifications from automatic deletes from lifecycle policies or from failed operations.
s3:ObjectRestore:Post *s3:ObjectRestore:Completed*	Using restore object event types you can receive notifications for initiation and completion when restoring objects from the S3 Glacier storage class. You use `s3:ObjectRestore:Post` to request notification of object restoration initiation. You use `s3:ObjectRestore:Completed` to request notification of restoration completion.
s3:ReducedRedundancyLostObject	You can use this event type to request Amazon S3 to send a notification message when Amazon S3 detects that an object of the RRS storage class is lost.
s3:Replication:OperationFailedReplication	You receive this notification event when an object that was eligible for replication using Amazon S3 Replication Time Control failed to replicate.
s3:Replication:OperationMissedThreshold	You receive this notification event when an object that was eligible for replication using Amazon S3 Replication Time Control exceeded the 15-minute threshold for replication.
s3:Replication:OperationReplicatedAfterThreshold	You receive this notification event for an object that was eligible for replication using the Amazon S3 Replication Time Control feature replicated after the 15-minute threshold.
s3:Replication:OperationNotTracked	You receive this notification event for an object that was eligible for replication using Amazon S3 Replication Time Control but is no longer tracked by replication metrics.

AWS - S3 Lifecycle Management

Lifecycle

- Lifecycle configuration enables you to specify the lifecycle management of objects in a bucket.
- The configuration is a set of one or more rules, where each rule defines an action for Amazon S3 to apply to a group of objects. These actions can be classified as follows:
- **Transition actions** – In which you define when objects transition to another storage class.
    - For example, you may choose to transition objects to the STANDARD_IA (IA, for infrequent access) storage class 30 days after creation, or archive objects to the GLACIER storage class one year after creation.
- **Expiration actions** – In which you specify when the objects expire. Then Amazon S3 deletes the expired objects on your behalf.

Reference:

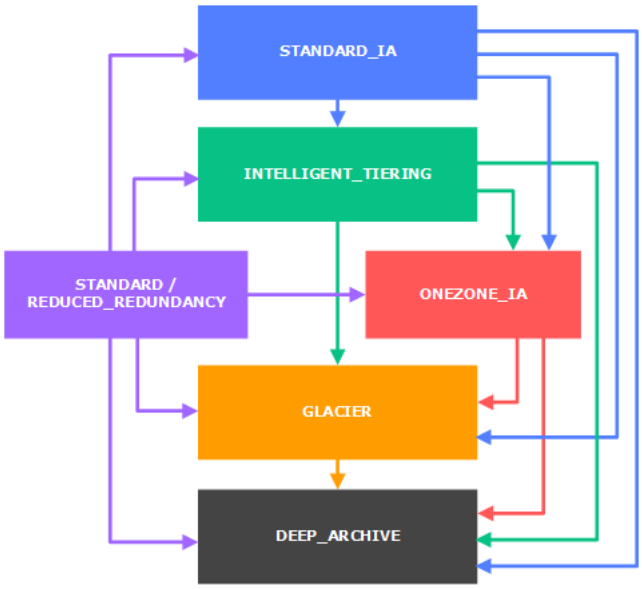
https://docs.aws.amazon.com/AmazonS3/latest/dev/object-lifecycle-mgmt.html

Check out this Amazon S3 Cheat Sheet:

https://tutorialsdojo.com/aws-cheat-sheet-amazon-s3/

Tutorials Dojo's AWS Certified Solutions Architect Associate Exam Study Guide:

https://tutorialsdojo.com/aws-cheat-sheet-aws-certified-solutions-architect-associate/



References:

https://aws.amazon.com/blogs/compute/amazon-s3-adds-prefix-and-suffix-filters-for-lambda-function-triggering

https://docs.aws.amazon.com/AmazonS3/latest/dev/object-lifecycle-mgmt.html

https://docs.aws.amazon.com/AmazonS3/latest/dev/lifecycle-configuration-examples.html

https://aws.amazon.com/s3/pricing

Check out this Amazon S3 Cheat Sheet:

https://tutorialsdojo.com/aws-cheat-sheet-amazon-s3/

S3 LIFE CYCLE management (across diff transition states)

- You can add rules in a lifecycle configuration to tell Amazon S3 to transition objects to another Amazon S3 storage class.
- For example:
  - When you know that objects are infrequently accessed, you might transition them to the STANDARD_IA storage class.
  - Or transition your data to the GLACIER storage class in case you want to archive objects that you don't need to access in real time.

- In a lifecycle configuration, you can define rules to transition objects from one storage class to another to save on storage costs.
- When you don't know the access patterns of your objects or your access patterns are changing over time, you can transition the objects to the INTELLIGENT_TIERING storage class for automatic cost savings.

- The lifecycle storage class transitions have a constraint when you want to transition from the STANDARD storage classes to either STANDARD_IA or ONEZONE_IA.
- **The following constraints apply:**
  - - For larger objects, there is a cost benefit for transitioning to STANDARD_IA or ONEZONE_IA.
  - Amazon S3 does not transition objects that are smaller than 128 KB to the STANDARD_IA or ONEZONE_IA storage classes because it's not cost effective.
  - - Objects must be stored **at least 30 days** in the current storage class before you can transition them to STANDARD_IA or ONEZONE_IA.
    - For example, you cannot create a lifecycle rule to transition objects to the STANDARD_IA storage class one day after you create them.
  - Amazon S3 doesn't transition objects within the first 30 days because newer objects are often accessed more frequently or deleted sooner than is suitable for STANDARD_IA or ONEZONE_IA storage.
  - If you are transitioning noncurrent objects (in versioned buckets), you can transition only objects that are at least 30 days noncurrent to STANDARD_IA or ONEZONE_IA storage.

- Since there is a time constraint in transitioning objects in S3, you can only change the storage class of your objects from S3 Standard storage class to STANDARD_IA or ONEZONE_IA storage after 30 days.

- This limitation does not apply on INTELLIGENT_TIERING, GLACIER, and DEEP_ARCHIVE storage class.

- In addition, the requirement says that the media assets should be fetched in a matter of minutes for a surprise **annual** data audit.
- This means that the retrieval will only happen once a year. You can use expedited retrievals in Glacier which will allow you to quickly access your data (within 1–5 minutes) when occasional urgent requests for a subset of archives are required.
- In this scenario, you can set a lifecycle policy in the bucket to transition to S3 - Standard IA after 30 days or alternatively, you can directly transition your data to Glacier after one week (7 days)

References:

https://docs.aws.amazon.com/AmazonS3/latest/dev/lifecycle-transition-general-considerations.html

https://docs.aws.amazon.com/AmazonS3/latest/dev/restoring-objects.html

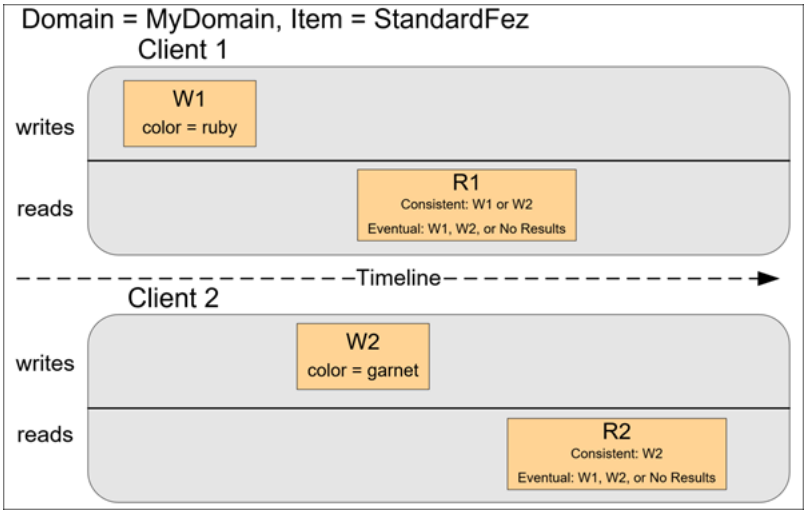
https://aws.amazon.com/s3/storage-classes/

Check out this Amazon S3 Cheat Sheet:

https://tutorialsdojo.com/aws-cheat-sheet-amazon-s3/

AWS - S3 Read Consistency

- Amazon S3 provides read-after-write consistency for PUTS of new objects in your S3 bucket in all regions with one caveat: if you make a HEAD or GET request to the key name (to find if the object exists) before creating the object, Amazon S3 provides eventual consistency for read-after-write.
- Amazon S3 offers eventual consistency for overwrite PUTS and DELETES in all regions.



- Updates to a single key are atomic.
- For example,
  - if you PUT to an existing key, a subsequent read might return the old data or the updated data, but it will never return corrupted or partial data.
  - This usually happens if your application is using parallel requests on the same object.

Amazon S3 achieves high availability by replicating data across multiple servers within Amazon's data centers.

If a PUT request is successful, your data is safely stored.

However, information about the changes must replicate across Amazon S3, which can take some time, and so you might observe the following behaviors:

- A process writes a new object to Amazon S3 and immediately lists keys within its bucket. Until the change is fully propagated, the object might not appear in the list.

- A process replaces an existing object and immediately attempts to read it. Until the change is fully propagated, Amazon S3 might return the prior data.

- A process deletes an existing object and immediately attempts to read it. Until the deletion is fully propagated, Amazon S3 might return the deleted data.

- A process deletes an existing object and immediately lists keys within its bucket. Until the deletion is fully propagated, Amazon S3 might list the deleted object.

- Amazon S3's support for parallel requests means you can scale your S3 performance by the factor of your compute cluster, without making any customizations to your application.
- Amazon S3 does not currently support Object Locking.
- If two PUT requests are simultaneously made to the same key, the request with the latest timestamp wins.
- If this is an issue, you will need to build an object-locking mechanism into your application.
- Updates are key-based; there is no way to make atomic updates across keys.
- For example, you cannot make the update of one key dependent on the update of another key unless you design this functionality into your application.

AWS - S3 Pre-signed URLs

References:

https://docs.aws.amazon.com/AmazonCloudFront/latest/DeveloperGuide/PrivateContent.html

https://docs.aws.amazon.com/AmazonS3/latest/dev/PresignedUrlUploadObject.html

Check out this Amazon CloudFront cheat sheet:

https://tutorialsdojo.com/aws-cheat-sheet-amazon-cloudfront/

S3 Pre-signed URLs vs CloudFront Signed URLs vs Origin Access Identity (OAI)

https://tutorialsdojo.com/aws-cheat-sheet-s3-pre-signed-urls-vs-cloudfront-signed-urls-vs-origin-access-identity-oai/

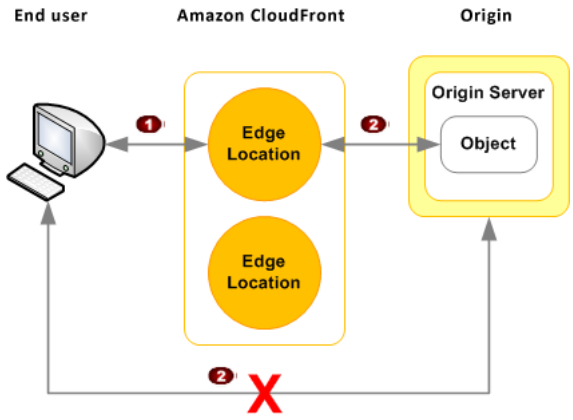
Comparison of AWS Services Cheat Sheets:

https://tutorialsdojo.com/comparison-of-aws-services-for-udemy-students/

CloudFront & AWS S3 Pre-signed URLs

- Many companies that distribute content over the Internet want to restrict access to documents, business data, media streams, or content that is intended for selected users, for example, users who have paid a fee.

- To securely serve this private content by using CloudFront, you can do the following:
  - - Require that your users access your private content by using special CloudFront signed URLs or signed cookies.
  - - Require that your users access your Amazon S3 content by using CloudFront URLs, not Amazon S3 URLs. Requiring CloudFront URLs isn't necessary, but it is recommended to prevent users from bypassing the restrictions that you specify in signed URLs or signed cookies. You can do this by setting up an origin access identity (OAI) for your Amazon S3 bucket. You can also configure the custom headers for a private HTTP server or an Amazon S3 bucket configured as a website endpoint.
  - All objects and buckets by default are private. The pre-signed URLs are useful if you want your user/customer to be able to upload a specific object to your bucket, but you don't require them to have AWS security credentials or permissions. You can generate a pre-signed URL programmatically using the AWS SDK for Java or the AWS SDK for .NET. If you are using Microsoft Visual Studio, you can also use AWS Explorer to generate a pre-signed object URL without writing any code. Anyone who receives a valid pre-signed URL can then programmatically upload an object.



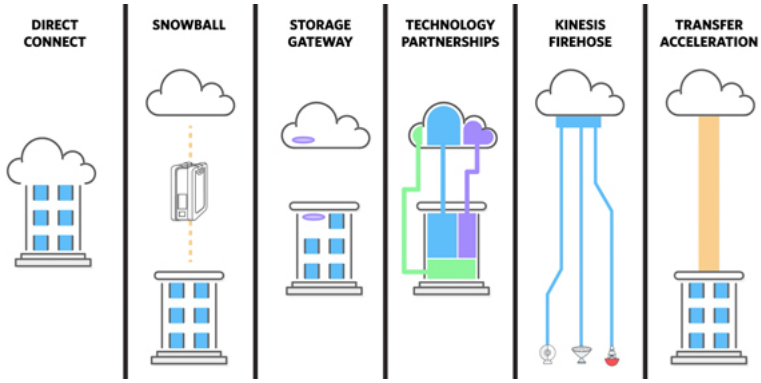
A global data analytics firm has various data centers from different countries all over the world. The staff are regularly uploading analytics, financial, and regulatory files of each of their respective data centers to a web portal deployed in AWS, which uses an S3 bucket named global-analytics-reports-bucket to durably store the data. The staff download various reports from a CloudFront distribution which uses the global-analytics-reports-bucket S3 bucket as the origin. You noticed that the staff are using both the CloudFront link and the direct Amazon S3 URLs to download the reports. The IT Security team of the company sees this as a security risk and they recommended that you implement a way to prevent anyone from bypassing CloudFront and using the direct Amazon S3 URLs. What would you do to meet the above requirement?

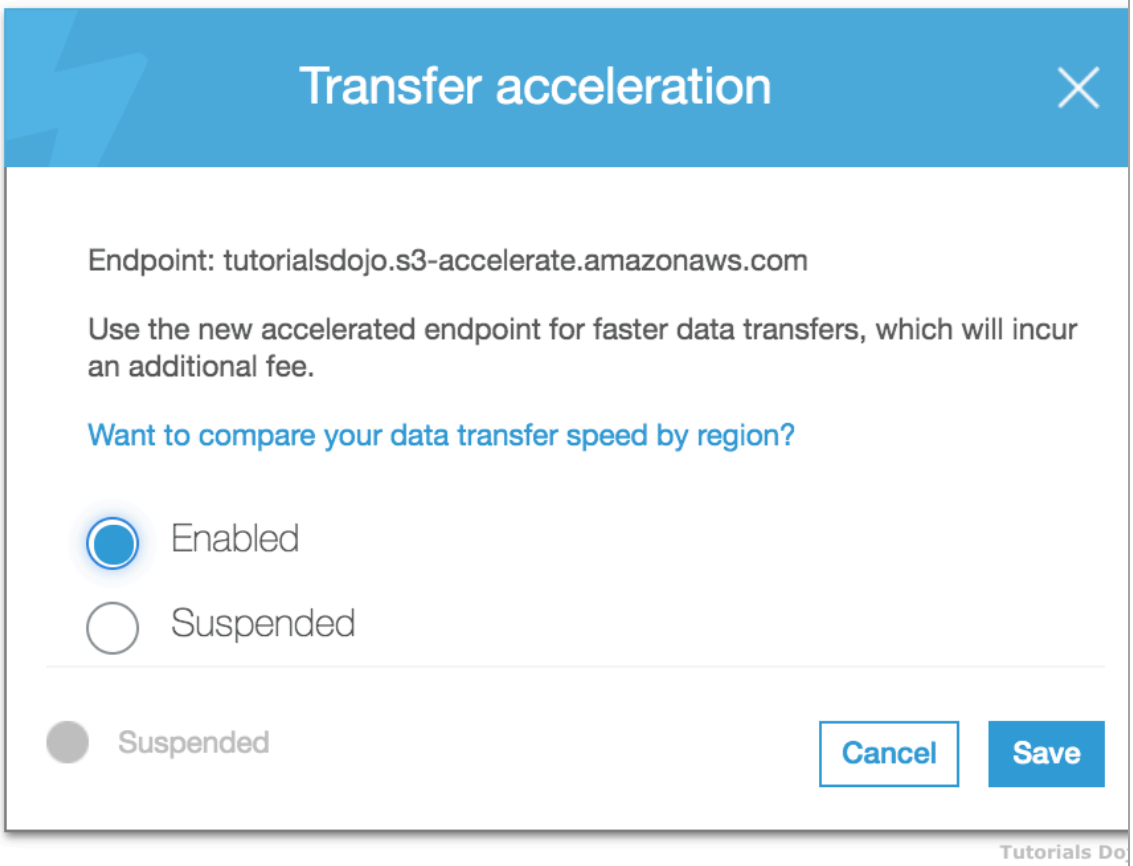
- 1. Create a special CloudFront user called an origin access identity (OAI) and associate it with your CloudFront distribution.
  2. Give the origin access identity permission to read the objects in your bucket.
  3. Remove anyone else's permission to use Amazon S3 URLs to read the objects.

AWS - S3 Transfer Acceleration

S3 Transfer Acceleration

- **Amazon S3 Transfer Acceleration enables fast, easy, and secure transfers of files over long distances between your client and your Amazon S3 bucket.**
- **Transfer Acceleration leverages Amazon CloudFront's globally distributed AWS Edge Locations.**
- **As data arrives at an AWS Edge Location, data is routed to your Amazon S3 bucket over an optimized network path.**





- **Amazon S3 Transfer Acceleration can speed up content transfers to and from Amazon S3 by as much as 50-500% for long-distance transfer of larger objects.**
- **Customers who have either web or mobile applications with widespread users or applications hosted far away from their S3 bucket can experience long and variable upload and download speeds over the Internet.**
- **S3 Transfer Acceleration (S3TA) reduces the variability in Internet routing, congestion and speeds that can affect transfers, and logically shortens the distance to S3 for remote applications.**
- **S3TA improves transfer performance by routing traffic through Amazon CloudFront's globally distributed Edge Locations and over AWS backbone networks, and by using network protocol optimizations.**

Reference:

http://docs.aws.amazon.com/AmazonS3/latest/dev/transfer-acceleration.html

Check out this Amazon S3 Cheat Sheet:

https://tutorialsdojo.com/aws-cheat-sheet-amazon-s3/

S3 Transfer Acceleration vs Direct Connect vs VPN vs Snowball vs Snowmobile:

https://tutorialsdojo.com/aws-cheat-sheet-s3-transfer-acceleration-vs-direct-connect-vs-vpn-vs-snowball-vs-snowmobile/

Comparison of AWS Services Cheat Sheets:

https://tutorialsdojo.com/comparison-of-aws-services-for-udemy-students/

AWS - S3 SELECT (SQL)

- With Amazon S3 Select, you can use simple structured query language (SQL) statements to filter the contents of Amazon S3 objects and retrieve just the subset of data that you need.
- By using Amazon S3 Select to filter this data, you can reduce the amount of data that Amazon S3 transfers, which reduces the cost and latency to retrieve this data.

- Amazon S3 Select works on objects stored in CSV, JSON, or Apache Parquet format.
- It also works with objects that are compressed with GZIP or BZIP2 (for CSV and JSON objects only), and server-side encrypted objects.
- You can specify the format of the results as either CSV or JSON, and you can determine how the records in the result are delimited

References:

https://docs.aws.amazon.com/AmazonS3/latest/dev/selecting-content-from-objects.html

https://docs.aws.amazon.com/redshift/latest/dg/c-using-spectrum.html

Check out these AWS Cheat Sheets:

https://tutorialsdojo.com/aws-cheat-sheet-amazon-s3/

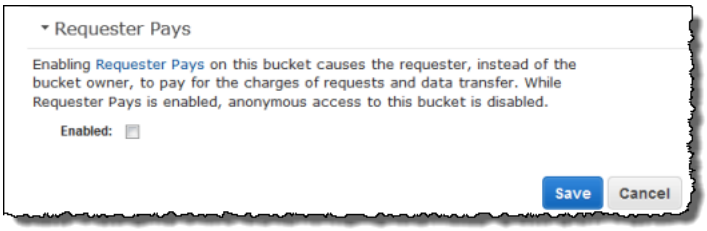
https://tutorialsdojo.com/aws-cheat-sheet-amazon-athena/

https://tutorialsdojo.com/aws-cheat-sheet-amazon-redshift/

AWS - S3 Request Payer

In general, bucket owners pay for all Amazon S3 storage and data transfer costs associated with their bucket. A bucket owner, however, can configure a bucket to be

a **Requester Pays**bucket. With Requester Pays buckets, the requester instead of the bucket owner pays the cost of the request and the data download from the bucket. The bucket owner always pays the cost of storing data.



You must authenticate all requests involving Requester Pays buckets. The request authentication enables Amazon S3 to identify and charge the requester for their use of the Requester Pays bucket. After you configure a bucket to be a Requester Pays bucket, requesters must include x-amz-request-payer in their requests either in the header, for POST, GET and HEAD requests, or as a parameter in a REST request to show that they understand that they will be charged for the request and the data download.

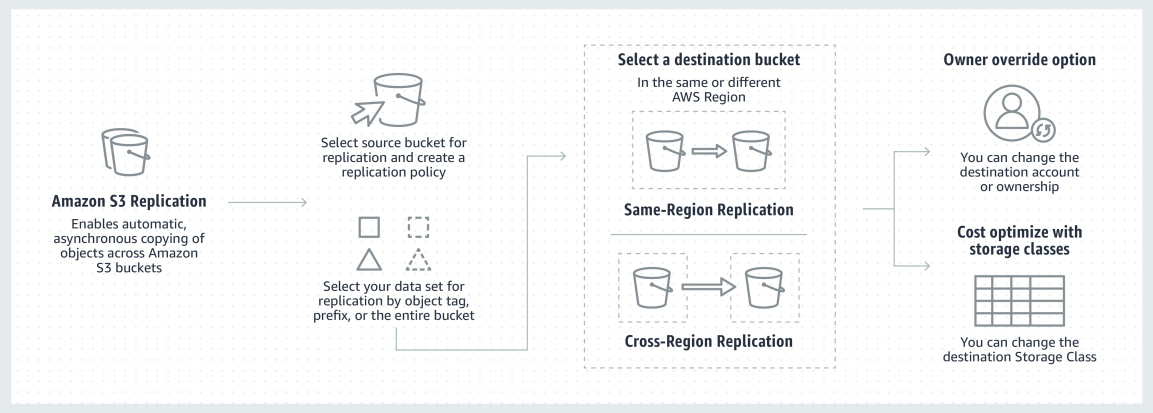
Reference:

https://docs.aws.amazon.com/AmazonS3/latest/dev/RequesterPaysBuckets.html

AWS - S3 Replication

S3 CROSS-REGION Replication

- Enable Cross-Region Replication to ensure that your S3 bucket would not be affected even if there is an outage in one of the Availability Zones or a regional service failure in us-east-1.
- When you upload your data in S3, your objects are redundantly stored on multiple devices across multiple facilities within the region only, where you created the bucket.
- Thus, if there is an outage on the entire region, your S3 bucket will be unavailable if you do not enable Cross-Region Replication, which should make your data available to another region.



Note that an Availability Zone (AZ) is more related with Amazon EC2 instances rather than Amazon S3 so if there is any outage in the AZ, the S3 bucket is usually not affected but only the EC2 instances deployed on that zone.

References:

https://aws.amazon.com/s3/faqs/

https://aws.amazon.com/s3/features/replication/

Check out this Amazon S3 Cheat Sheet:

https://tutorialsdojo.com/aws-cheat-sheet-amazon-s3/

AWS - Athena

- Amazon Athena is an interactive query service that makes it easy to analyze data in Amazon S3 using standard SQL expressions.
- Athena is serverless, so there is no infrastructure to manage, and you pay only for the queries you run. Athena is easy to use.
- Simply point to your data in Amazon S3, define the schema, and start querying using standard SQL expressions.
- Most results are delivered within seconds.
- With Athena, there's no need for complex ETL jobs to prepare your data for analysis.
- This makes it easy for anyone with SQL skills to quickly analyze large-scale datasets.

Reference:

https://aws.amazon.com/s3/features/#Query_in_Place

Check out these AWS Cheat Sheets:

https://tutorialsdojo.com/aws-cheat-sheet-amazon-s3/

https://tutorialsdojo.com/aws-cheat-sheet-amazon-athena/

https://tutorialsdojo.com/aws-cheat-sheet-amazon-redshift/