Curso
Herramientas de Computación en la Nube

Clase 4

Febrero 14 de 2026

Almacenamiento de Objetos y Arquitecturas tipo S3

Profesor

Alvaro Mauricio Montenegro Díaz, Ph.D.
Universidad de la Sabana

Febrero 2026


Almacenamiento de Objetos y Arquitecturas tipo S3
Data Lakes y Lakehouse
Pipelines
Híbrido
ETL / ML
Consistencia y Semántica
• Strong Read-after-Write
• Objetos Inmutables
• No OLTP.
Durabilidad y Disponibilidad
• Erasure Coding
• Replicación Multi-AZ
• Autosanación
S3
Bucket
API
Seguridad y Gobernanza
• IAM, KMS, Encryption
• Versioning Object Lock.
Modelo Conceptual
Portabilidad y Escala
• Compatibilidad de facto
Bucket
Objetos
Metadatos
Versionid
API
• Standard
• IA / Glacier
• Deep Archive
Tiering & Costo / Performance
• Standard
• IA / Glacier
• Deep Archive

Introducción general

El almacenamiento constituye uno de los pilares fundamentales de cualquier sistema de información. En arquitecturas modernas orientadas a la nube, el almacenamiento deja de concebirse como un simple subsistema de archivos y pasa a convertirse en un **servicio distribuido, altamente durable y desacoplado del cómputo**.

En este contexto emerge el **almacenamiento de objetos**, cuyo representante paradigmático es S3. Este modelo no solo redefine cómo se almacenan los datos, sino también **cómo se diseñan las arquitecturas de software, los flujos de datos, los sistemas analíticos y las plataformas de aprendizaje automático**.

En esta lección estudia S3 **no como un producto**, sino como **un modelo arquitectónico**.

1. ¿Qué es almacenamiento de objetos?

El almacenamiento de objetos organiza la información como **entidades autocontenidas**, llamadas objetos, cada una compuesta por:

- un **payload binario** (los datos),
- un conjunto de **metadatos**,
- y una **clave única** dentro de un contenedor lógico.

A diferencia de los sistemas de archivos tradicionales:

- no existen directorios reales,
- no hay jerarquía física,
- no hay operaciones POSIX,
- y no existe noción de “archivo abierto”.

Todo acceso se realiza **mediante una API** y una **clave lógica**.

2. S3 como arquetipo de diseño

S3 (Simple Storage Service) es el sistema que **cristaliza este paradigma** y lo lleva a escala planetaria. Su importancia no radica únicamente en su implementación, sino en que define:

- un **modelo de consistencia claro**,
- una **semántica de objetos inmutables**,
- una separación radical entre **almacenamiento y cómputo**,
- y una interfaz que se convierte en **estándar de facto**.

Hoy, cuando se habla de almacenamiento en la nube, en realidad se habla de **arquitecturas compatibles con S3**.

3. Modelo conceptual de S3

El modelo de S3 es deliberadamente simple:

- **Bucket**: contenedor lógico globalmente único.
- **Objeto**: unidad atómica de almacenamiento.
- **Key**: identificador único (string).
- **Metadata**: información asociada al objeto.
- **VersionId** (opcional): control de versiones.

No existen:

- joins,
- transacciones multiobjeto,
- índices relacionales,
- ni referencias internas.

Esta simplicidad es la clave de su **escalabilidad extrema**.

4. Prefijos, no directorios

En S3 no hay carpetas reales. Lo que aparenta ser una estructura jerárquica es únicamente un **prefijo de texto**:

```
s3://data-lake/silver/clientes/2026/01/clientes.parquet
```

Toda la “ruta” es una **clave plana**.

Esto tiene consecuencias arquitectónicas profundas:

- renombrar es costoso (copiar + borrar),
- mover datos no es una operación trivial,
- el diseño de prefijos es parte del **diseño del sistema**, no un detalle operativo.

5. Consistencia y semántica de acceso

S3 ofrece **consistencia fuerte read-after-write** para:

- creación de nuevos objetos,
- sobrescritura,
- borrado.

Esto permite razonar correctamente sobre pipelines modernos. Sin embargo, **no hay garantías transaccionales entre múltiples objetos**, lo que obliga a diseñar flujos idempotentes y reproducibles.

6. Durabilidad y disponibilidad

S3 está diseñado bajo un principio clave:

> *La durabilidad es más importante que la latencia: 99.999999999% (11 nueves)*

Con durabilidades del orden de **11 nueves**, S3 logra esto mediante:

- replicación multi-zona,
- codificación de borrado,
- mecanismos automáticos de reparación.

Disponibilidad y durabilidad no son equivalentes, y esta distinción es central en arquitecturas críticas.

7. Clases de almacenamiento y ciclo de vida

S3 separa **API** de **características físicas** mediante clases de almacenamiento:

- acceso frecuente,
- acceso infrecuente,
- archivo,

- archivo profundo.

Las **políticas de ciclo de vida** permiten que los datos evolucionen automáticamente a través de estas clases sin modificar las aplicaciones.

Esto convierte al almacenamiento en una **variable económica controlable por diseño**.

8. Seguridad, gobernanza y control

S3 incorpora un modelo de seguridad multinivel:

- políticas de identidad,
- políticas sobre recursos,
- cifrado en reposo,
- control de acceso público,
- versionado,
- retención legal (WORM).

Este conjunto permite cumplir requisitos regulatorios estrictos (finanzas, justicia, auditoría, ciencia).

9. S3 no es una base de datos

Un punto crítico de esta lección es comprender **qué no es S3**:

- no es OLTP,
- no es un filesystem,
- no es adecuado para baja latencia,
- no soporta locking ni concurrencia fina.

Por diseño, S3 se integra con **motores externos** que aportan:

- consultas,
- transacciones,
- análisis,
- inferencia.

10. S3 como base del Data Lake moderno

S3 es la **capa de persistencia** de arquitecturas modernas:

- Data Lakes
- Lakehouse
- ETL / ELT
- Feature Stores
- ML pipelines

Los motores de consulta (Spark, Trino, DuckDB) viven **encima**, no dentro del almacenamiento.

Esto materializa el principio:

> *Storage is cheap, compute is ephemeral.*

11. Mini-sistema ilustrativo: Data Lake simplificado

Dominio mínimo:

- Datos crudos (raw)
- Datos procesados (bronze / silver)
- Datos analíticos (gold)

Organización conceptual en S3:

```
s3://empresa-datalake/raw/
s3://empresa-datalake/bronze/
s3://empresa-datalake/silver/
s3://empresa-datalake/gold/
```

Cada capa representa una **decisión arquitectónica**, no una carpeta.

12. Lenguajes y acceso a S3

S3 no se consulta directamente. Se accede mediante:

- APIs (PUT / GET / DELETE),

- SDKs,
- frameworks analíticos,
- motores SQL externos.

La consulta ocurre **fuera** del almacenamiento, reforzando la separación de responsabilidades.

13. Convergencia y estandarización

Hoy, múltiples sistemas implementan el mismo modelo conceptual:

- soluciones cloud,
- soluciones on-premise,
- soluciones híbridas.

Diseñar pensando en S3 significa diseñar para **portabilidad, longevidad y escalabilidad**.

14. Síntesis conceptual

No existe un sistema de almacenamiento universalmente superior. El almacenamiento de objetos tipo S3 es óptimo cuando:

- los datos son grandes,
- la durabilidad es crítica,
- el acceso es masivo,
- el cómputo puede desacoplarse.

La decisión arquitectónica clave no es **qué tecnología usar**, sino **qué responsabilidades asignar a cada capa**.

Cierre de la lección

S3 no debe entenderse como un producto de nube, sino como una **abstracción fundacional** sobre la cual se construyen los sistemas de datos modernos.

Comprender este modelo es comprender **la nube misma**.

15. Diccionario de términos técnicos

Almacenamiento de objetos (Object Storage)

Modelo de almacenamiento que gestiona los datos como objetos autocontenidos, cada uno identificado por una clave única y acompañado de metadatos. Prioriza escalabilidad, durabilidad y simplicidad sobre semánticas complejas de acceso.

Objeto (Object)

Unidad atómica de almacenamiento compuesta por datos binarios, metadatos y una clave. Los objetos son inmutables desde el punto de vista lógico: una modificación implica crear una nueva versión.

Bucket

Contenedor lógico de objetos. Define el ámbito de nombres, políticas de seguridad, versionado y reglas de ciclo de vida. No equivale a un directorio.

Key (clave de objeto)

Identificador único de un objeto dentro de un bucket. Es una cadena de texto plana; las aparentes “carpetas” son prefijos convencionales.

Prefijo (Prefix)

Convención semántica usada para agrupar objetos mediante claves con inicio común. No representa una estructura jerárquica física.

API de almacenamiento de objetos

Interfaz basada en operaciones simples (PUT, GET, DELETE, LIST) que desacopla el acceso lógico a los datos de su organización física.

Consistencia fuerte (Strong Read-after-Write Consistency)

Garantía según la cual una escritura o eliminación es inmediatamente visible para las lecturas posteriores. No implica transacciones multiobjeto.

Inmutabilidad lógica

Principio por el cual los objetos no se modifican “en sitio”. Las actualizaciones se modelan como nuevas versiones.

Versioning (Control de versiones)

Mecanismo que conserva versiones históricas de un objeto, permitiendo recuperación, auditoría y gobernanza de datos.

Durabilidad

Probabilidad de que un objeto no se pierda en el tiempo. En sistemas tipo S3 se logra mediante replicación, codificación de borrado y autosanación.

Disponibilidad

Capacidad del sistema para responder a solicitudes en un momento dado. Es un concepto distinto de durabilidad.

Erasure Coding

Técnica de protección de datos que divide la información en fragmentos redundantes distribuidos, reduciendo el costo frente a la replicación completa.

Multi-AZ (Multi–Availability Zone)

Estrategia de despliegue que replica datos entre zonas físicamente separadas para tolerancia a fallos.

Clases de almacenamiento (Storage Classes)

Configuraciones que equilibran costo, latencia y disponibilidad según el patrón de acceso esperado.

Lifecycle Policy (Política de ciclo de vida)

Reglas automáticas que migran objetos entre clases de almacenamiento o los eliminan según antigüedad o uso.

Data Lake

Arquitectura que utiliza almacenamiento de objetos como repositorio central de datos crudos y procesados, desacoplado del cómputo.

Lakehouse

Enfoque que combina Data Lake con estructuras de gestión de esquema, versiones y transacciones a nivel de archivos.

Separación almacenamiento–cómputo

Principio arquitectónico que desacopla la persistencia de datos del procesamiento, permitiendo escalabilidad independiente.

Persistencia políglota

Uso de múltiples tecnologías de almacenamiento en una misma arquitectura, seleccionadas según patrón de acceso y semántica requerida.

OLTP (Online Transaction Processing)

Tipo de carga caracterizada por transacciones cortas, frecuentes y de baja latencia. No es un caso de uso para S3.

WORM (Write Once, Read Many)

Modelo de retención que impide la modificación o eliminación de datos durante un periodo definido, común en entornos regulatorios.

S3-compatible

Implementación que expone la misma semántica y API que S3, independientemente del proveedor o despliegue (cloud u on-premise).

16. Bibliografía

Libros (fundamentales)

1. **Designing Data-Intensive Applications.** Martin Kleppmann. O’Reilly Media, 2017. Referencia central para comprender almacenamiento distribuido, inmutabilidad, consistencia y separación storage/compute.
2. **Cloud Native Data Infrastructure.** Matt Stine et al.. O’Reilly Media, 2023. Arquitecturas modernas de datos en la nube, object storage y data platforms.

3. **Building Data-Intensive Applications on the Cloud.** Andreas Kretz. O’Reilly Media, 2022. Enfoque práctico sobre data lakes, S3, ETL y sistemas analíticos cloud-native.
4. **Designing Cloud Data Platforms.** Danil Zburivsky. O’Reilly Media, 2024. Diseño de plataformas modernas basadas en almacenamiento de objetos.

Artículos y documentos técnicos

5. **Amazon S3 – System Design Overview.** Amazon Web Services Whitepaper. Documento oficial que describe durabilidad, consistencia y modelo operativo de S3.
6. **The Snowflake Elastic Data Warehouse.** Dageville et al., SIGMOD 2016. Caso canónico de separación almacenamiento–cómputo sobre object storage.
7. **Delta Lake: High-Performance ACID Table Storage over Cloud Object Stores.** Armbrust et al., VLDB 2020. Demuestra cómo se construyen semánticas avanzadas *encima* de S3.

Referencias conceptuales complementarias

8. **Site Reliability Engineering.** Google. O’Reilly Media. Para comprender disponibilidad, durabilidad y trade-offs operativos.
9. **Fundamentals of Data Engineering.** Joe Reis & Matt Housley. O’Reilly, 2022. Marco moderno para entender data lakes, pipelines y arquitecturas de datos.