Лекція 3. Введення у сховища даних, Blob об’єкти.

Тема 1. Опис концепцій та типів сховищ даних: Blob-об’єкти, Файли, Черги, Таблиці.

Тема 2. Рішення для забезпечення довговічності даних: локально надлишкове сховище (LRS), зонально надлишкове сховище (ZRS), географічно надлишкове сховище (GRS — Geo-redundant storage), а також варіанти з кількома регіонами.

Тема 3. Рівні доступу до даних: Hot, Cool, Archive.

Тема 4. Механізми зворотного видалення (soft delete) великих об’єктів.

Тема 5. Інтерфейси для взаємодії з Azure Blob Storage.

Сховища даних

Сховище Azure — це хмарне рішення для зберігання сучасних додатків, що забезпечує високу доступність і масштабованість для задоволення актуальних потреб.

Уявіть собі його як головний контейнер або інструментальну скриньку. Це один, захищений ресурс у вашій підписці Azure, який надає унікальний простір імен для зберігання та доступу до даних. Коли ви створюєте обліковий запис сховища, ви отримуєте не один тип сховища, а універсальний ресурс, який може містити **чотири фундаментально різні сервіси даних**. Саме тому я називаю його «швейцарським ножем» у світі даних Azure — тут є інструмент на будь-який випадок."

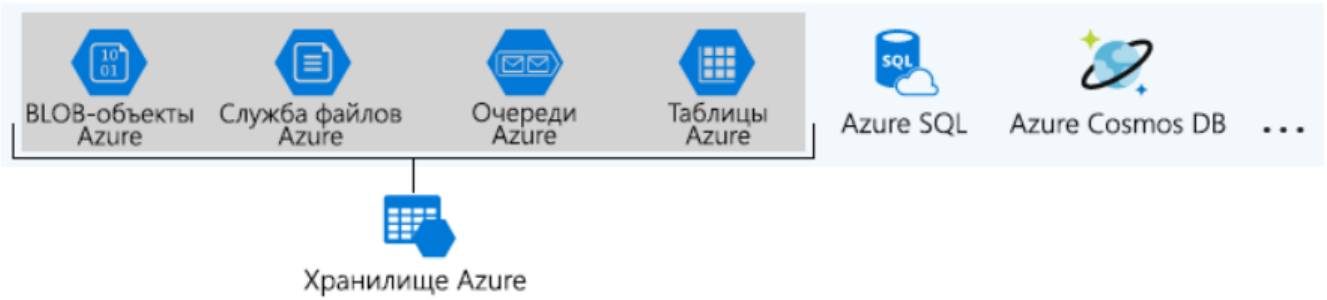
Azure об'єднує чотири служби даних, що називаються "Служби сховища Azure":

1.BLOB-об'єкти Azure – сховище для великих текстових і двійкових файлів, тобто неструктурованих даних.

2.Файли Azure – сховище для зберігання файлів.

3.Черги Azure – сховище для зберігання повідомлень.

4.Таблиці Azure – сховище для напівструктурованих даних.



soloveistoragedemo | Storage browser

Storage account

Search

Overview
Activity log
Tags
Diagnose and solve problems
Access Control (IAM)
Data migration
Events
Storage browser
Storage Mover
Data storage
Security + networking
Networking

soloveistoragedemo
Favorites
Recently viewed
Blob containers
File shares
Queues
Tables

Privacy settings | Feedback

Storage account metrics

The data provided is regularly updated about 2-4 times a day and published hourly. If your account has extremely la
objects, it may be over a day between updates.

Blob containers

Number of containers	-
Number of blobs	-
Total data stored	-

File shares

Number of file shares	-
Number of files	-
Total data stored	-

Tables

Number of tables	-
Number of entities	-
Total data stored	-

Queues

Number of queues	-
Number of messages	-
Total data stored	-

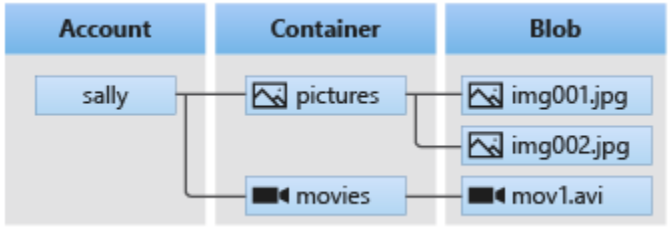
Тип сховища	Аналогія	Використовувати, коли потрібно...	Приклад
Blob	Шафа для файлів	Зберігати файли, зображення, відео, резервні копії.	Хостинг ресурсів вебсайту.
File	Мережевий диск	Переносити додатки, мати спільний диск.	Спільні конфігурації для ВМ.
Queue	Список справ	Роз'єднати додатки, обробляти роботу пізніше.	Черга обробки зображень.
Table	Каталог карток	Зберігати дані без схеми з швидким пошуком за ключем.	Дані профілю користувача.

Azure Blob Storage — це рішення для зберігання великих обсягів неструктурованих даних у хмарі Microsoft Azure. Воно дозволяє зберігати дані у вигляді об'єктів (blobs), таких як текстові або двійкові файли. Це популярне рішення для створення резервних копій, архівів, зберігання мультимедійних даних, а також для обробки аналітичних великих даних.

Основні компоненти Azure Blob Storage:
Storage Account (обліковий запис зберігання):
Це логічна одиниця, яка надає доступ до Azure Storage. Обліковий запис є контейнером для різних типів сховищ, включаючи Blob Storage.
Обліковий запис зберігання може використовуватися для створення і керування різними типами сховищ даних: Blob, Queue, File і Table Storage.
Він надає контроль доступу, моніторинг і управління для збережених даних.
Containers (контейнери):

Контейнер є логічним екземпляром у сховищі Blob Storage, що містить групу Blob-об'єктів.
Кожен обліковий запис зберігання може містити кілька контейнерів, а кожен контейнер — необмежену кількість об'єктів (blobs).
Blobs (об'єкти):

- Це безпосередньо дані, які зберігаються в контейнері. Існує три типи blob-об'єктів:
- Block blobs: зберігають текст або двійкові дані, які ідеально підходять для зберігання великих файлів, таких як документи або мультимедіа.
- Append blobs: це варіант block blob, який призначений для запису даних у послідовному порядку, наприклад, для лог-файлів.
- Page blobs: використовуються для зберігання даних у вигляді сторінок і підходять для сценаріїв з випадковим доступом, таких як віртуальні жорсткі диски (VHD).



Основні характеристики кожного облікового запису сховища

"Незалежно від того, яким сервісом ви користуєтеся в межах облікового запису, ви отримаєте потужні вбудовані функції:"

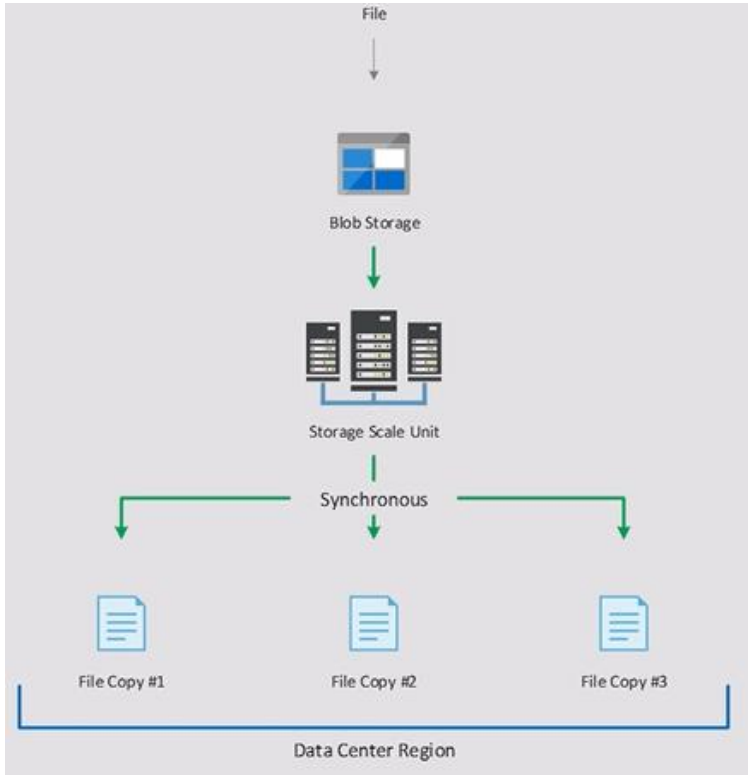
•**Надійність і висока доступність:**
Ваші дані автоматично реплікуються. З **локально-резервованим сховищем (LRS)** Azure зберігає три копії ваших даних в одному дата-центрі. З **гео-резервованим сховищем (GRS)** дані дублюються в іншому регіоні за сотні кілометрів, щоб захистити вас у разі регіонального збою.

•**Безпека:**
Доступ контролюється за допомогою кількох механізмів: ключі доступу, **спільні підписи доступу (SAS)**, сучасна **рольова модель доступу (RBAC)**, інтегрована з Azure Active Directory. Дані шифруються як у стані спокою, так і під час передачі.

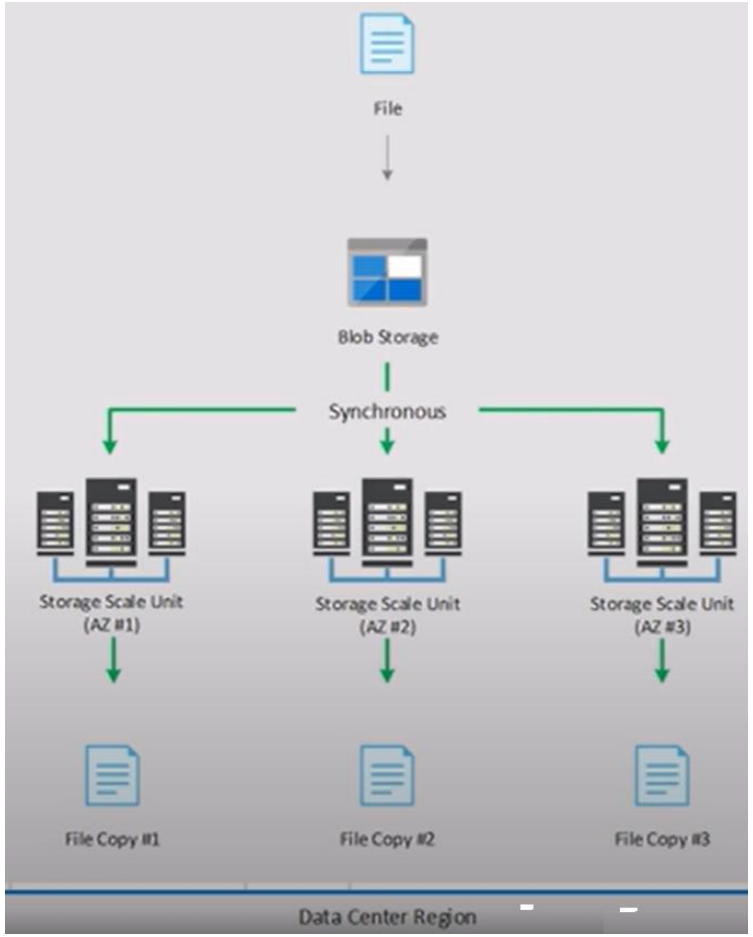
•**Масштабованість:**
Облікові записи сховищ розраховані на величезні обсяги даних. Один акаунт може зберігати до **5 петабайтів** (тобто 5000 терабайтів) інформації.

•**Доступність:**
Дані доступні з будь-якої точки світу через HTTP/HTTPS за допомогою **REST API**. Це означає, що будь-яка мова програмування або інструмент може взаємодіяти зі сховищем.

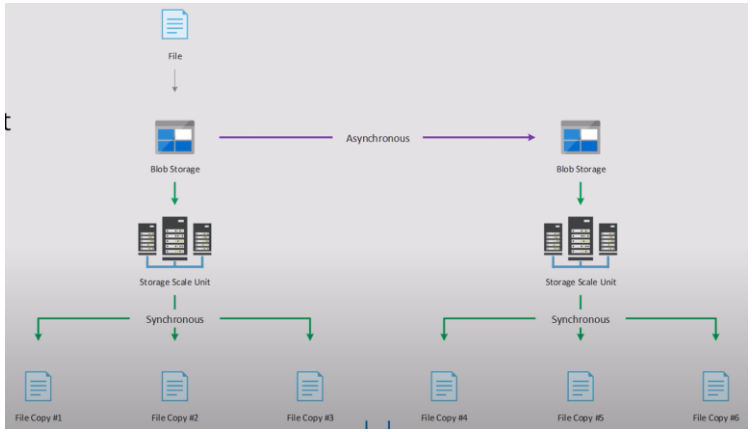
- Отже, коли ми говоримо про **довговічність** (durability), ми ставимо основне питання: Що станеться, якщо апаратне забезпечення, на якому зберігаються мої дані, вийде з ладу? Диск може зламатися, сервер може втратити живлення або може статися проблема з усім дата-центром.
- Рішення Azure просте, але потужне: **ніколи не зберігайте тільки одну копію**.
- За замовчуванням кожен файл, який ви записуєте в **Azure Storage Account**, автоматично реплікується кілька разів. Як і де ці копії зберігаються, визначається вибраною вами опцією редундації при створенні акаунта. Давайте розглянемо три основні стратегії, починаючи з найпростішої.



Рівень 1 – LRS - Локально надлишкові дані (LRS) – дані копіюються в одному сховище.
Azure створює три копії ваших даних і зберігає їх у межах одного фізичного дата-центру. Ці копії розподілені по різних апаратних стійках для захисту від збоїв диска або стійки сервера.
Аналогія: Ви пишете важливий звіт, друкуєте його тричі і кладете кожну копію в різну шафу для документів в одному офісному приміщенні.
Захищає від:
Поломки диска
Поломки стійки сервера
Вразливе до:
Збої на рівні дата-центру (наприклад, пожежа, повінь, повне відключення електроенергії). Якщо весь будинок вийде з ладу, всі три копії будуть втрачені.
Найкраще для: Неважливі дані, середовище для розробки/тестування або дані, які можна легко відновити.



- **Рівень 2 - ZRS (**Надлишкові дані зони **)**
- "Наступний рівень — Надлишкові дані зони **(ZRS)**. Зона доступності — це фізично окремий дата-центр у межах одного регіону Azure, з незалежним живленням, охолодженням і мережею."
- • **Що це таке:** Azure створює три копії ваших даних і розподіляє їх по трьох різних зонах доступності в межах основного регіону.
• **Аналогія:** Ви пишете звіт і кладете одну копію в бібліотеку, одну — в корпус наук і одну — в корпус мистецтв — всі в одному університетському кампусі.
• **Захищає від:**
• Поломка апаратного забезпечення (диск, стійка)
• Поломка цілого дата-центру
• **Вразливе до:**
• Великомасштабні збої регіону (наприклад, великий землетрус або ураган, що впливає на весь географічний регіон).
• **Найкраще для:** Додатків з високою доступністю, які не можуть дозволити собі зупинку через поломку одного дата-центру.



- Рівень 3 - Надлишкові дані в декількох регіонах. «GRS - Geo-redundant storage» – дані копіюються асинхронно в регіоні «дублере», але синхронно в файли регіону.
- Ця стратегія призначена для відновлення після катастроф.
- Що це таке: Надлишкові дані в декількох регіонах копіює ваші дані до вторинного регіону, що знаходиться сотні або навіть тисячі миль від основного регіону. Це означає, що навіть якщо природна катастрофа виведе з ладу цілий регіон, ваші дані залишаються в безпеці.
- Аналогія: Ви надсилаєте копію своєї дисертації на зберігання довіреному родичу, що живе в іншій країні.
- Два варіанти: Azure пропонує два основні типи гео-надлишковості:
- GRS (Geo-Redundant Storage): Стандартна опція.
- GZRS (Geo-Zone-Redundant Storage): Преміум-опція, яка поєднує переваги ZRS і GRS.

LRS (Локально резервоване зберігання):
Данні зберігаються в межах одного дата-центру, що захищає від апаратних збоїв у межах цього центру.

GRS (Гео-резервоване зберігання):
Дані зберігаються в основному центрі даних і автоматично реплікуються в інший географічно віддалений центр для захисту від серйозних аварій.

RA-GRS (Читання доступу до гео-резервованого зберігання):
Те ж саме, що і GRS, але з можливістю читання даних навіть у разі збою основного центру.

ZRS (Зональне резервування):
Дані зберігаються в декількох зонах в межах одного регіону для забезпечення більшої доступності.

GZRS (Гео-зональне резервування):
Реплікація даних по декількох географічних зонах, що підвищує доступність і відмовостійкість.

GRS (Geo-Redundant Storage)	**GZRS (Geo-Zone-Redundant Storage)**
1. LRS в основному регіоні (3 копії в одному дата-центрі). 2. Копії реплікуються в вторинний регіон, де також використовується LRS (3 копії в одному дата-центрі). **3. Загальна кількість копій:** 6 (3 в основному, 3 в вторинному)	ZRS в основному регіоні (3 копії в 3 зонах). Копії реплікуються в вторинний регіон, який використовує LRS (3 копії в одному дата-центрі). **Загальна кількість копій:** 6 (3 в зонах основного регіону, 3 в вторинному)
Забезпечує: Відновлення після катастроф	**Забезпечує:** Високу доступність І відновлення після катастроф
Найкраще для: Захисту від регіональних відмов.	**Найкраще для:** Критичних додатків, які потребують захисту як від збоїв дата-центру, так і від регіональних катастроф.

Blob Storage (Неструктуровані дані)

"Перший і найпоширеніший сервіс — це **Blob Storage**. 'Blob' означає **Binary Large Object**, тобто великий двійковий об'єкт — простіше кажучи, файл. Це місце для зберігання **неструктурованих даних**."

Що таке неструктуровані дані?

Усе, що не має чіткої схеми: зображення, відео, аудіо, лог-файли, резервні копії, документи, віртуальні диски.

Основні сценарії використання:

•Подача зображень і документів напряму в браузер.

•Зберігання файлів для розподіленого доступу (наприклад, інсталятори програм).

•Стримінг відео й аудіо.

•Резервне копіювання, аварійне відновлення, архівація.

Ключові поняття:

•**Container (контейнер):** Папка для організації blob-об'єктів.

•**Blob:** Сам файл.

•**Рівні доступу (Hot, Cool, Archive):** Для оптимізації вартості.

- **Hot** — часто доступні дані.
- **Cool** — для рідко використовуваних.
- **Archive** — для довгострокового зберігання (низька ціна, але з затримкою доступу).

Рівні доступу (Hot, Cool, Archive):

Уявіть склад. Найбільш часто використовувані предмети зберігаються біля входу, щоб їх було легко забрати. Менш часто використовувані предмети знаходяться в задній кімнаті. Архівні записи відправляються в захищений, віддалений сховище. **Azure Blob Storage** працює саме так, допомагаючи вам заощаджувати кошти."

Рівні доступу — це налаштування для ваших blob-об'єктів, які збалансовують витрати на зберігання та витрати/час на доступ до даних.

- **Правило:** Чим дешевше зберігати дані, тим дорожче і/або повільніше буде до них отримати доступ.
- **Мета:** Підібрати найекономічніший рівень доступу, що відповідає вашому шаблону використання даних.

Характеристика	Hot Tier	Cool Tier	Archive Tier
Призначено для	Активні дані, файли, що використовуються	Резервні копії, менш нові дані	Довгострокове архівування, дані для відповідності нормативам
Доступність	99.9%	99%	Н/Д (офлайн)
Затримка при відновленні	Мілісекунди	Мілісекунди	Години (через відновлення)
Вартість зберігання	Найвища	Нижча	Найнижча
Вартість доступу	Найнижча	Вища	Найвища
Мінімальний термін зберігання	Немає	30 днів	180 днів
Приклад використання	Зображення для активного вебсайту	Місячні резервні копії фінансових даних	Зберігання юридичних документів на 7 років

Тип облікового запису	Ресурси зберігання	Типи резервного копіювання
Стандартний обліковий запис загального призначення v2	Blob Storage (включаючи Data Lake Storage1), Queue Storage, Table Storage та Azure Files	Локально резервоване зберігання (LRS) / гео-резервоване зберігання (GRS) / читання доступу до гео-резервованого зберігання (RA-GRS)
Зонально-резервоване зберігання (ZRS) / гео-зонально-резервоване зберігання (GZRS) / читання доступу до гео-зонально-резервованого зберігання (RA-GZRS)2	Стандартний тип облікового запису для blob-об'єктів, файлових ресурсів, черг та таблиць. Рекомендується для більшості сценаріїв використання Azure Storage. Якщо потрібна підтримка мережевої файлової системи (NFS) у Azure Files, використовуйте тип облікового запису преміум для файлових ресурсів.	
Преміум block blobs3	**Blob Storage (включаючи Data Lake Storage1)**	LRS
ZRS2	Преміум тип облікового запису для block blob-об'єктів та append blob-об'єктів. Рекомендується для сценаріїв з високими транзакційними навантаженнями або тих, що використовують менші об'єкти або потребують стабільно низької затримки зберігання. Дізнайтесь більше про приклади робочих навантажень.	
Преміум файлові ресурси3	**Azure Files**	LRS
ZRS2	Преміум тип облікового запису тільки для файлових ресурсів. Рекомендується для підприємств або високопродуктивних масштабованих додатків. Використовуйте цей тип облікового запису, якщо хочете мати обліковий запис з підтримкою як SMB, так і NFS файлових ресурсів.	
Преміум page blobs3	Тільки page blobs	LRS
ZRS2	Преміум тип облікового запису тільки для page blob-об'єктів. Дізнайтесь більше про page blob-об'єкти та приклади використання.	

Стандартний обліковий запис загального призначення v2

Ресурси:

Blob Storage (включаючи Data Lake Storage), Queue Storage, Table Storage, Azure Files.

Опис:

Це універсальний тип облікового запису для зберігання різних типів даних у Azure. Він підтримує всі типи даних, включаючи блокові об'єкти (blob), черги (queue), таблиці (table) і файлові ресурси (Azure Files).

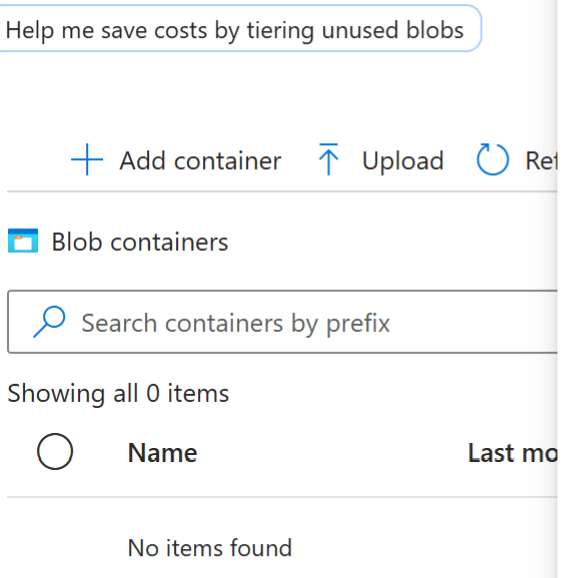
Резервне копіювання:

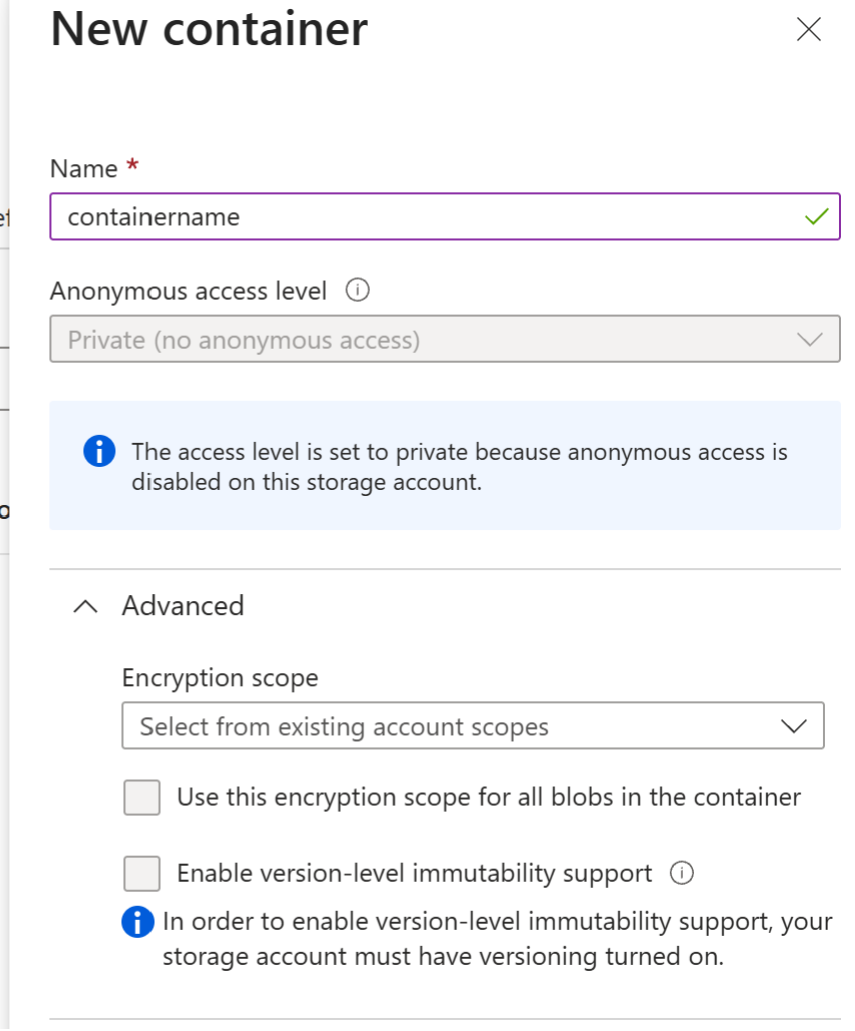
Підтримує локально резервоване зберігання (LRS), гео-резервоване зберігання (GRS), читання доступу до гео-резервованого зберігання (RA-GRS) — це різні стратегії резервного копіювання для забезпечення доступності та відмовостійкості даних.

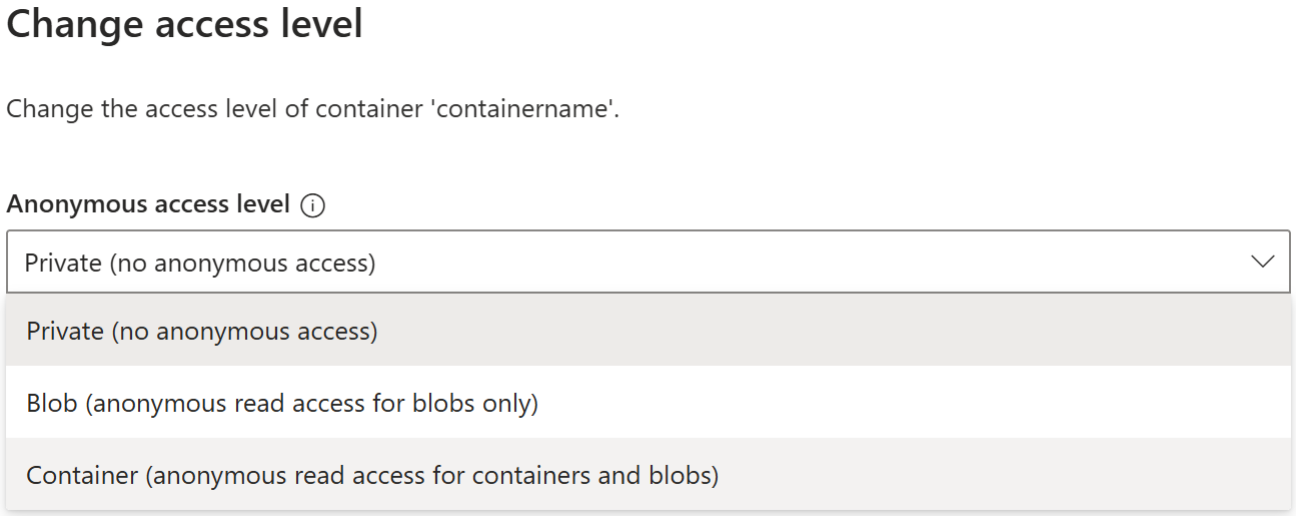
Конфігурації

Workload	***Account kind***	***Performance***	***Redundancy***	***Hierarchical namespace enabled***	***Default access tier***	***Soft delete enabled***
Cloud native	General purpose v2	Standard	ZRS, RA-GRS	No	Hot	Yes
Analytics	General purpose v2	Standard	ZRS1, RA-GRS	Yes2	Hot	Yes
High performance computing (HPC)	General purpose v2	Standard	ZRS, RA-GRS	Yes	Hot	Yes
Backup and archive	General purpose v2	Standard	ZRS, RA-GRS	No	Cool3	Yes
Machine learning and artificial intelligence	General purpose v2	Standard	ZRS, RA-GRS	Yes	Hot	No

За замовчуванням усі дані контейнера є приватними.
Доступ мають лише власник облікового запису або авторизовані користувачі.
Це захищає від несанкціонованого доступу до конфіденційних файлів.
Blob (Блоб):
Дозволяє анонімний доступ для читання окремих блобів. Потрібна URL-адреса блоба. Приклад: доступ до зображення напряму.
Container (Контейнер): Дозволяє анонімний читання + перегляд списку усіх блобів у контейнері.Користувачі можуть переглядати вміст.







Розглянемо приклад доступу до файлу з рівнем доступу **Архівне зберігання**

Showing all 1 items

Name	Last modified	Access tier	Blob type	Size	Lease state
index.html	9/10/2024, 2:43:55 PM	Archive	Block blob	48 B	Available

You do not have access

Blob: index.html

This request is not authorized to perform this operation using this permission.

Summary

Session ID
4b9744a64bc84c049c9fd8c7a20a710c

Resource ID
/subscriptions/601a84f6-579c-4ba6-ac84-00c597a85bd...

Extension
Microsoft_Azure_Storage

Content
BlobPropertiesBladeV2

Error code
403

Storage Request ID
5d747c16-b01e-0071-0481-0305c1000000

Details

- This request is not authorized to perform this operation using this permission. RequestId:5d747c16-b01e-0071-0481-0305c1000000 Time:2024-09-10T12:58:32.6882236Z
- Learn more about authorizing access to Azure Storage

Azure Storage Account – Властивості

Огляд ключових властивостей Blob, File, Queue, Table, Security, Networking

Blob Storage: Hierarchical namespace

- Disabled
- Дозволяє файлову ієрархію (Data Lake Gen2)
- За замовчуванням вимкнено

Hierarchical Namespace (ієрархічний простір імен) — це можливість у Azure Data Lake Storage Gen2, яка дозволяє працювати з Blob Storage як із файловою системою:

підтримуються каталоги та підкаталоги;

можна керувати доступом до окремих папок;

ефективні операції з файлами (переміщення, видалення директорій і т.д.).

```
/project1/
   └── report1.csv
/project1/data/
   └── datafile1.csv
/project2/
   └── summary.docx
```

З HNS можемо:

1. Видалити всю папку project1/ разом з усіма файлами всередині.
2. Призначити окремі права доступу для /project1/data/.
3. Перемістити файл summary.docx з project2/ у project1/.

Blob Storage: Default access tier

- Hot – часто використовувані дані
- Cool – рідко використовувані
- Archive – архівні дані

Blob Storage: Blob anonymous access

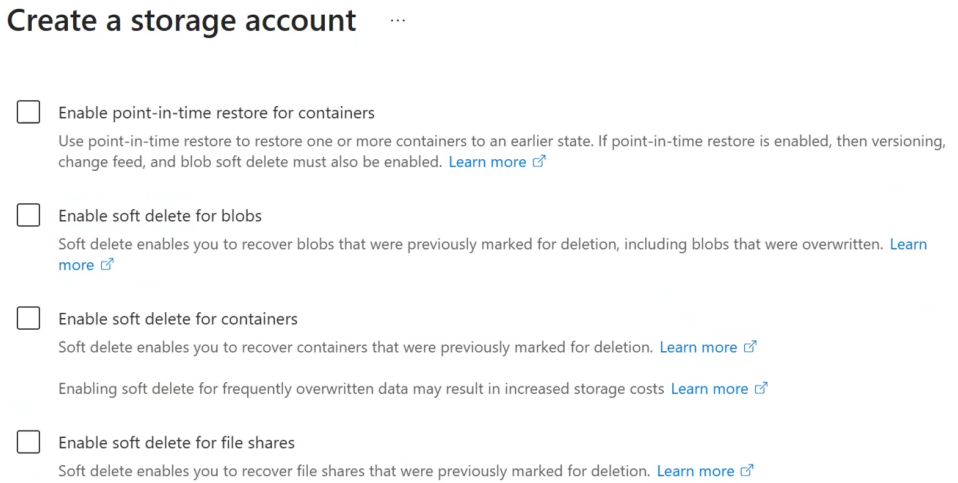
- Disabled
- Вимкнено за замовчуванням
- Включення робить контейнери публічними

Blob Storage: Blob soft delete

- Disabled
- Аналог кошика
- Дозволяє відновити випадково видалені об’єкти

Проблема без **soft delete**
Видалений блоб за замовчуванням зникає назавжди.
Неможливо відновити випадково видалені дані.
Ризики втрати критично важливої інформації.
Що таке **Soft Delete**?
Функція захисту від випадкового видалення.
Замість остаточного видалення блоб переміщується у стан soft delete.
Можна відновити протягом визначеного періоду (Retention period).
Як це працює
Користувач видаляє блоб.
Блоб зберігається у прихованому стані.
Протягом N днів (Retention policy) його можна відновити.
Після завершення періоду – блоб видаляється остаточно.

Приклад
Увімкнено soft delete з політикою збереження 7 днів.
Студент видалив файл report.docx.
Протягом 7 днів викладач може відновити його.
Після 7 днів файл буде видалено безповоротно.



Blob Storage: Versioning

- Disabled
- Дозволяє зберігати попередні версії файлів

Blob Storage: Change feed

- Disabled
- Журнал змін усіх операцій
- Change Feed — це журнал подій, який автоматично записує всі зміни у контейнерах Blob Storage:
- створення blob'a,
- зміна/перезапис blob'a,
- видалення blob’a.

Сценарії

1. Аудит змін у системі зберігання - Потрібно знати, коли і хто змінив звіт.

```
{
  "eventType": "BlobCreated",
  "blobUrl": "https://myaccount.blob.core.windows.net/reports/monthly/report-sep.csv",
  "timestamp": "2025-09-15T10:23:00Z"
}
```

Синхронізація з базою даних - Change Feed дозволяє слухати зміни й оновлювати відповідний запис у базі: Новий blob → додати запис, Видалення → видалити запис, Оновлення → змінити статус

Blob Storage: NFS v3

- Disabled
- Доступ до Blob як до мережевої файлової системи

Функція	**Без HNS** (звичайний Blob Storage)	**З HNS** (Data Lake Gen2)
Папки	Лише імітація через імена blob'ів	Реальні папки
Видалення каталогу	Потрібно видаляти всі файли вручну	Один запит — видаляє все
Копіювання/переміщення файлів	Через копіювання + видалення	Підтримується нативно
ACL (доступ до окремих директорій)	Не підтримується	Підтримується

Сценарії:
Якщо потрібно зберігати великі об'єми структурованих даних.
Якщо потрібне тонке управління доступом до папок.

Blob Storage: Cross-tenant replication

- Disabled
- Реплікація між різними Azure AD tenants

Cross-tenant replication дозволяє автоматично реплікувати (копіювати) дані з одного облікового запису Azure Storage до іншого, навіть якщо вони належать до різних Azure Active Directory (AAD) tenant-ів або Azure підписок.

Це важливо для:

1. розподілу даних між організаціями,
2. бекапу в інші середовища (наприклад, disaster recovery),
3. партнерських інтеграцій, де одна організація надає дані іншій.

Сценарії:

1. Між двома компаніями
Сценарій: Компанія А збирає телеметрію зі своїх IoT-пристроїв у Blob Storage.
Компанія В (аналітичний партнер) має власний тенант і хоче обробляти ці дані у своєму середовищі.

Приклад через Azure CLI

```
az storage account or-policy create \
    --account-name mystorageaccountsrc \
    --resource-group myrg \
    --destination-account mystorageaccountdest \
    --rules name=rule1 \
    --source-container container1 \
    --destination-container container2
```

Blob Storage: Storage tasks assignments

- None
- Керування доступами і завданнями адміністрування

Storage tasks assignments — це механізм, який дозволяє призначати певні завдання або ролі користувачам, сервісам або скриптам для роботи з Azure Blob Storage, використовуючи рольову модель доступу (RBAC), Managed Identities або SAS-токени.

Приклади

Призначення завдання: Завантаження файлів у контейнер

Призначити роль: Storage Blob Data Contributor

Доступ: тільки до потрібного контейнера

```
az role assignment create \
  --role "Storage Blob Data Contributor" \
  --assignee user@domain.com \
  --scope
"/subscriptions/xxxx/resourceGroups/myrg/providers/Microsoft.Storage/storageAccounts/mystorage/blo
bServices/default/containers/mycontainer"
```

Типові ролі для Blob Storage

Призначення завдання: Тільки читання файлів	Storage Blob Data Reader
Автоматичне резервне копіювання Видалення старих файлів (архівування) Завантаження, редагування, видалення blob'ів	Storage Blob Data Contributor
Повний контроль (включно з ACL)	Storage Blob Data Owner
Якщо сервіс використовує черги разом з blob'ами	Storage Queue Data Contributor

Security: Require secure transfer

- Enabled
- Вимагає HTTPS для REST API. Усі запити мають використовувати HTTPS. HTTP-запити автоматично відхиляються.

5. Шифрування за замовчуванням

- Усі дані шифруються автоматично при створенні Storage Account.
- Використовуються ключі, керовані Microsoft:
- Microsoft створює, зберігає та обслуговує ключі.
- Користувач не керує ними самостійно.

6. Користувацькі ключі (Customer-managed keys, CMK)

- Для повного контролю можна використовувати власні ключі через Azure Key Vault:
- Створення ключа.
- Налаштування політик доступу.
- Можливість заміни чи відкликання ключа.

Інтерфейси для взаємодії з Azure Blob Storage

DefaultAzureCredential — це клас в Azure SDK, який автоматично підбирає найбільш відповідний метод автентифікації для вашого середовища.
DefaultAzureCredential реалізує ланцюжок автентифікації (chained credentials), в якому пробує кілька способів автентифікації по черзі, поки один з них не спрацює.

Порядок перевірки

1. EnvironmentCredential — змінні середовища (ENV)
2. ManagedIdentityCredential — для Azure ресурсів (наприклад, VM, App Service)
3. SharedTokenCacheCredential — кеш токенів Azure CLI / Visual Studio
4. VisualStudioCredential — акаунт, залогінений у VS
5. VisualStudioCodeCredential — акаунт у VS Code
6. AzureCliCredential — акаунт, залогінений через Azure CLI
7. AzurePowerShellCredential — акаунт, залогінений у PowerShell
8. InteractiveBrowserCredential (опційно) — відкриття браузера для автентифікації

Основні методи DefaultAzureCredential- GetToken(TokenRequestContext requestContext)
Основний метод інтерфейсу TokenCredential.
Використовується для отримання AccessToken.
Параметр: TokenRequestContext — містить список scope (областей доступу).
Повертає: AccessToken

Інтерфейси для взаємодії з Azure Blob Storage

Клас BlobServiceClient є основним класом для роботи з Azure Blob Storage в Python за допомогою бібліотеки azure-storage-blob. Він дозволяє виконувати операції на рівні сервісу блобів, такі як створення контейнерів, доступ до блобів, а також керування доступом та властивостями контейнерів. Цей клас є частиною більшої бібліотеки, яка надає інтерфейси для взаємодії з Azure Blob Storage.

Основні функціональні можливості класу BlobServiceClient:

- З'єднання з сервісом Azure Blob Storage.
- Створення, управління та видалення контейнерів.
- Отримання списку контейнерів та блобів.
- Аутентифікація та управління доступом.
- Управління політиками шифрування та надійності.

Ініціалізація класу BlobServiceClient:

Для того, щоб почати роботу з BlobServiceClient, необхідно створити екземпляр цього класу, передавши URL вашого аккаунта Azure Blob Storage та обрану форму аутентифікації. Найпоширенішими способами аутентифікації є використання connection string або Azure Identity. Спосіб -

```
from azure.storage.blob import BlobServiceClient
from azure.identity import DefaultAzureCredential
account_url = "https://<your_account_name>.blob.core.windows.net"

credential = DefaultAzureCredential()
blob_service_client = BlobServiceClient(account_url=account_url, credential=credential)
```

```
from azure.storage.blob import BlobServiceClient
connection_string = "<your_connection_string>"
blob_service_client = BlobServiceClient.from_connection_string(connection_string)
```

Основні методи класу BlobServiceClient

1. Створення контейнера

```
container_name = "my-container"
container_client = blob_service_client.create_container(container_name)
```

Отримання списку контейнерів	containers = blob_service_client.list_containers() for container in containers: print(container['name'])
Видалення контейнера	blob_service_client.delete_container(container_name)

Create a container asynchronously

Це означає асинхронне створення контейнера в Azure Blob Storage за допомогою async/await в Python.

Асинхронність дозволяє не блокувати програму під час очікування відповіді від Azure.

Це корисно:

- ✓ при роботі з багатьма контейнерами або blob'ами одночасно;
- ✓ у веб-додатках;

Рядок	Пояснення
await blob_service_client.create_container(...)	Асинхронне створення контейнера
async def ... + await	Асинхронна функція
asyncio.run(...)	Запуск асинхронного коду в Python

```
import asyncio
from azure.storage.blob.aio import BlobServiceClient
from azure.core.exceptions import ResourceExistsError
async def create_container_async():
conn_str =
"DefaultEndpointsProtocol=https;AccountName=your_account;AccountKey=your_key;EndpointSuffix=core.windows.net"
 blob_service_client = BlobServiceClient.from_connection_string(conn_str)
 container_name = "my-async-container"
 try:
        # Створення контейнера
        container_client = await blob_service_client.create_container(container_name)

    except ResourceExistsError:
        print(f" Контейнер '{container_name}' вже існує.")
    finally:
        await blob_service_client.close()

asyncio.run(create_container_async())
```

Клас BlobClient є частиною бібліотеки azure-storage-blob і використовується для виконання операцій з окремими блобами (файлами) в Azure Blob Storage. Він надає методи для взаємодії з окремими об'єктами блобів, включаючи завантаження, читання, оновлення та видалення файлів, а також управління їх метаданими, властивостями та доступом.

Основні можливості класу BlobClient:

- ✓ Завантаження та зчитування даних з блобів.
- ✓ Завантаження метаданих та властивостей блобів.
- ✓ Оновлення даних або властивостей блоба.
- ✓ Видалення блобів.
- ✓ Управління доступом через SAS токени або інші механізми.

Створення об'єкта BlobClient	blob_client = ***BlobClient***(account_url=account_url, container_name=container_name, blob_name=blob_name, credential=credential) blob_client = ***BlobClient.from_connection_string***(connection_string, container_name=container_name, blob_name=blob_name)

Для завантаження файлу до Azure Blob Storage використовуємо метод **upload_blob()** Параметри методу upload_blob: data: Дані, які потрібно завантажити (наприклад, відкритий файл). overwrite: Якщо встановлено в True, існуючий блоб буде перезаписано.	from azure.storage.blob import BlobClient blob_client = BlobClient(account_url=account_url, container_name=container_name, blob_name=blob_name, credential=credential) with open("local_file.txt", "rb") as data: blob_client.***upload_blob***(data, overwrite=True)
Для зчитування блоба з Azure Storage використовується метод **download_blob()**. Він повертає об'єкт, який можна використовувати для читання даних. **readall()**: Зчитує всі дані блоба. Зазвичай цей метод використовується для невеликих файлів.	with open("downloaded_blob.txt", "wb") as download_file: download_file.write(blob_client.***download_blob***().readall())
Властивості блоба (наприклад, content type) можна змінювати за допомогою методу **set_blob_properties()**.	blob_client.***set_blob_properties***(content_type="text/plain")

Delete and restore blobs	blob_client.delete_blob()
	deleted_blobs =
	blob_service_client.**get_container_client**(container_name).list_blobs(include=["deleted"])
	
	for deleted_blob in deleted_blobs:
	  if deleted_blob.name == blob_name:
	  blob_service_client.undelete_blob(container_name, blob_name)
	  print("Blob відновлено")

Find blob using tags	blob_client.**set_**blob_tags({'project': 'ai', 'owner': 'dev’}) tag_filter = "\"project\"='ai'" blobs = blob_service_client.find_blobs_by_tags(tag_filter)
Manage blob leases Lease корисний для **захисту blob'ів від змін** (наприклад, у багатопотокових програмах).	lease = blob_client.acquire_lease() lease.renew() lease.break_lease()
Manage blob properties and metadata	metadata = {"author": "Andrii", "version": "1.2"} blob_client.set_blob_metadata(metadata)
Set or change a blob access tier	blob_client.set_standard_blob_tier("Cool")

Azure Storage Explorer

https://azure.microsoft.com/en-au/products/storage/storage-explorer?utm_source=chatgpt.com

Питання

1. Які основні відмінності між Blob Storage, File Storage, Queue Storage та Table Storage в Azure?
2. Що таке blob-об’єкт і які типи blob’ів існують?
3. У яких випадках доцільно використовувати Azure File Storage замість Blob Storage?
4. Для яких сценаріїв призначені Queue Storage та Table Storage?
5. Які типові бізнес-задачі можна вирішити за допомогою Block blobs, Append blobs та Page blobs?
6. У чому полягає різниця між локально надлишковим сховищем (LRS) та зонально надлишковим сховищем (ZRS)?
7. Які переваги надає географічно надлишкове сховище (GRS) порівняно з LRS та ZRS?
8. Чому варто враховувати затримки та вартість при виборі варіанту копіювання даних між регіонами?
9. У яких випадках GRS може бути критично важливим для бізнесу?
10. Чим відрізняються рівні доступу Hot, Cool і Archive за вартістю та продуктивністю?
11. Які критерії використовуються для вибору відповідного рівня доступу до даних?
12. Що відбувається з даними, коли їх переводять у рівень Archive, і які обмеження це створює?
13. Що таке «soft delete» у Azure Blob Storage і як воно захищає дані від випадкового видалення?
14. Чим відрізняється soft delete для blob'ів, контейнерів та версій об’єктів?
15. Які інтерфейси й інструменти можна використовувати для роботи з Azure Blob Storage (наприклад, REST API, SDK, Azure CLI, PowerShell) та які їхні переваги?