Université de Sherbrooke

Systèmes de gestion de bases de données

Stockage

UdeS:SGBD_01

Christina KHNAISSER *(christina.khnaisser@usherbrooke.ca)*

Luc LAVOIE *(luc.lavoie@usherbrooke.ca)*

(les auteurs sont cités en ordre alphabétique nominal)

—

CoFELI/Scriptorum/SGBD_01-Stockage, version 1.1.0.a, en date du 2025-09-30

— document de travail, ne pas citer —

Sommaire

Introduction aux techniques de stockage de données massives utilisées dans les systèmes de gestion de bases de données.

Mise en garde

Le présent document est en cours d'élaboration ; en conséquence, il est incomplet et peut contenir des erreurs.

Historique

diffusion	**resp.**	**description**
2025-09-30	LL	Revue préalable aux activités 2025-3.
2025-01-15	LL	Revue préalable aux activités 2025-1.
2024-09-16	CK	Récupération de notes diverses.
2024-08-16	LL	Ébauche initiale à partir de documents antérieurs produits entre 2004 et 2023.

Table des matières

Introduction

Au sein d'un SGBD, l'organisation du stockage des données et l'ensemble des algorithmes d'accès aux données forment le modèle physique de données (selon l'architecture trischématique).

Le problème du stockage est celui de la conciliation des nombreuses caractéristiques des dispositifs de stockage à choisir, assembler et exploiter afin de garantir de façon

- **exacte** (strictement cohérente, valide et efficace),
- **pratique** (suffisamment complète, efficiente et évolutive)
- **pérenne** (dans la mesure des moyens à disposition)

le stockage et l'accès à des données organisées selon un modèle logique déterminé.

Le présent module a pour but de présenter une synthèse des techniques de stockage de données utilisées par les SGBD. La présentation repose sur une connaissance des bases de fonctionnement des systèmes d'exploitation et des systèmes de gestion de fichiers contemporains.

Contenu des sections

- La section 1 expose la problématique générale du stockage des données en faisant ressortir le grand nombre de caractéristiques à considérer et la variété des solutions, ce qui justifie son abstraction dans une couche spécifique (la couche physique) au sein de l'architecture trischématique.
- La section 2 présente un survol de plusieurs technologies contemporaines, confirmant la problématique exposée à la section 1 et utilisées dans les exemples des hiérarchies de stockage classique (section 3) et contemporaine (section 7). Elle justifie également la nécessité de développer un modèle de stockage indépendant de l'interface de la couche physique.
- La section 3 présente les hiérarchies classiques des systèmes établis et courants.
- La section 4 rappelle certains principes organisationnels des fichiers qui seront utilisés pour présenter les interfaces d'accès et les mécanismes de palliation aux défaillances à la section 6 ainsi que les mécanismes de pagination à la section 5.
- La section 5 (à venir) présentera les fonctionnalités de la pagination et certaines techniques courantes.
- La section 6 (à venir) présentera les techniques de palliation aux défaillances.
- La section 7 (à venir) présentera les hiérarchies de stockage utilisées par certains systèmes de pointe.
- La section 8 (à venir) reprendra les différents critères de décision utilisés dans les sections précédentes et en dégagera les principales mesures utiles.

Le sujet de l'indexation, étroitement lié aux différents sujets traités dans le présent module, est traité dans un module distinct `SGBD_02-Indexation`.

Sources du document

La première version du document a été établie sur la base des travaux publiés par Delobel, Elmasri, Galvin, Navathe, Snodgrass, Silberschatz, Tanenbaum et Ullman.

Compléments au document

- [Elmasri2016a], chapitres 16 et 17
- [Hainaut2022], chapitre 4
- [Lelarge2023a], chapitres 2, 3 et 5
- [Petrov2019], chapitre 1
- [Sciore2020a], chapitres 3, 4, 6 et 7

1. Technologies

Trois catégories de dispositifs de stockage sont présentées :

- électro-mécaniques,
- électroniques,
- autres.

Les dispositifs de stockage les plus couramment utilisés se classent dans les deux premières catégories.

1.1. Dispositifs électro-mécaniques

C'est-à-dire tout dispositif dont le stockage et les fonctions d'accès ne dépendent que de composantes électroniques et mécaniques (au moins une électronique et une mécanique).

1.1.1. Exemples contemporains

Disque « magnétique »

Exploration autonome, voir références.

Disque « laser »

Exploration autonome, voir références.

Ruban « magnétique » (en cassette)

Exploration autonome, voir références.

1.1.2. Exemples historiques

Ruban perforé

Exploration autonome, voir références.

Carte perforée

Exploration autonome, voir références.

Tambour « magnétique »

Exploration autonome, voir références.

Ruban « magnétique » (en bobine)

Exploration autonome, voir références.

1.2. Dispositifs électroniques

C'est-à-dire tout dispositif dont le stockage et les fonctions d'accès ne dépendent que de composantes électroniques.

1.2.1. Exemples contemporains

Quatre types de module mémoire sont présentés en ordre décroissant de performance (latence et débit)… et de cout (unité monétaire par unité de volume).

MM0 (module mémoire de niveau 0)

Mémoire DDR SDRAM typiquement intégrée aux circuits de la puce hébergeant le processeur selon l'une des configurations suivantes :

- répartie entre les coeurs de calcul (donc sans contention),
- commune entre eux (donc avec contention possible),

- hybride (généralement avec peu ou pas de contention).

Ce type de mémoire est caractérisée par un accès uniforme direct sans latence ni variance notable entre les emplacements.

MM1 (module mémoire de niveau 1)

Mémoire (SDR|DDR) (SDRAM|DDRAM) typiquement organisée en barrettes et accessible au processeur par l'entremise d'un bus **interne**.

Accès uniforme direct avec très faible latence et très faible variance entre les emplacements. Lorsque le bus est partagé, en fonction du trafic, la contention peut augmenter significativement la latence et la variabilité.

MM2 (module mémoire de niveau 2)

Mémoire DRAM ou NVRAM typiquement organisée en barrettes et accessible au processeur par l'entremise d'un bus **externe**.

Accès uniforme direct avec faible latence et faible variance entre les emplacements Lorsque le bus est partagé, en fonction du trafic, la contention peut augmenter significativement la latence et la variabilité.

MM3 ((module mémoire de niveau 3, solid state drive, SSD)

Mémoire NVRAM organisée en dispositif indépendant et accessible au processeur par l'entremise d'un bus **externe**.

Le SSD utilise les mêmes technologies de base que le MM2 doté de NVRAM, mais offre une rémanence intrinsèque. Il est généralement conditionné comme un dispositif indépendant, ayant sa propre alimentation électrique et utilisant les mêmes canaux et protocoles que les disques électro-mécaniques.

1.3. Autres dispositifs

Plusieurs autres technologies sont utilisées (mémoires optiques) ou en émergence (mémoires ADN), mais non présentées dans le cadre du présent module.

2. Hiérarchie classique des stockages

Les données d'une base de données sont stockées physiquement grâce à un dispositif de stockage. Il n'est toutefois généralement pas souhaitable, voire possible, de n'utiliser qu'un seul type de dispositif.

Pour les opérations en temps réel (ou quasi réel), on voudra privilégier la vitesse. Par contre, le cout plus élevé et les capacités limitées de ces dispositifs amènent le plus souvent à recourir à des dispositifs moins chers, à plus grande capacité. Par ailleurs, la rémanence est une propriété importante qui doit être assurée à long terme grâce au journal de transaction, aux copies de sécurité et aux archives (le « D » de ACID). Pour cela, d'autres dispositifs stables, à grande capacité et peu couteux sont requis.

Organisation

Typiquement, au moins trois niveaux de dispositifs seront mis à contribution, parfois deux ou quatre. Il faut donc prévoir des mécanismes de transfert entre ces niveaux (typiquement la pagination, la journalisation, la duplication et la réplication).

Discussion

Quelles sont les caractéristiques distinguant les journaux, les copies de sécurité et les archives ?

Suite

Les trois prochaines sous-sections décrivent une hiérarchie classique en trois niveaux.

2.1. Stockage primaire

C'est le dispositif devant pouvoir être exploité directement par l'unité de traitement (*central processing unit*

CPU) de l'ordinateur. On distingue deux sous-niveaux :

- La **mémoire cache** (historiquement SRAM, désormais MM0) requise pour l'exécution des instructions et la composition des résultats.
- La **mémoire principale** (historiquement SRAM, désormais MM1) pour sauvegarder les instructions elles-mêmes et les données requises par une transaction.

Le stockage primaire permet généralement un accès rapide aux données, mais sa capacité de stockage est limitée et le contenu peut être perdu en cas de coupure de courant ou de panne du système.

2.2. Stockage secondaire

C'est le dispositif qui permet de stocker en ligne l'ensemble des données de l'état courant de la base de données.

Les technologies le plus souvent utilisées sont le disque électro-mécanique et le SSD.

2.3. Stockage tertiaire

On distingue deux sous-niveaux de stockage tertiaire selon qu'il doit impérativement être proximal ou distant (hors site). On distingue deux sous-niveaux :

- Le **stockage proximal** est principalement destiné au journal principal (dont une copie pourra (devrait) toutefois être stockée hors site et dès lors traité comme une copie de sécurité). Le journal doit être maintenu proximal puisqu'il est susceptible d'être sollicité en cours du traitement transactionnel, lorsqu'une transaction est annulée. Typiquement les dispositifs doivent être accessibles via un bus externe et les technologies SSD et disque magnétique sont privilégiées.
- Le **stockage distant** est requis pour les copies de sécurité et les archives. Typiquement les dispositifs seront accédés via un réseau de communication (en raison de la distance présumée inter-sites) et les dispositifs électro-mécaniques sont privilégiés en raison de leur cout plus faible que les dispositifs électroniques, pour un même volume de données.

2.4. Tendances

De nos jours, plusieurs ordinateurs peuvent être dotés d'une grande capacité en mémoire primaire, de sorte qu'il devient possible d'y conserver la totalité de la base de données. On parle alors de base de données « en mémoire » (*in memory database*).

Dans ce cas, des copies de sécurité plus fréquentes, appelées instantanés (*snapshot*), sont requise afin de pallier les pannes sans avoir à « rejouer » le journal sur une trop longue durée. Ces instantanés sont généralement conservés en mémoire secondaire ou tertiaire.

2.5. La suite

La grande variété des technologies et des niveaux de stockage requis par le SGBD est susceptible d'entrainer une grande complexité de mise en oeuvre (comportant un paramétrage important variant dynamiquement en fonction de la nature des requêtes et des conditions d'exploitation) au risque de transparaitre dans les modèles logiques et conceptuels.

Pour éviter cela, un modèle physique, ajoutant une couche d'abstraction au modèle de stockage, doit être développé et sera présenté dans les deux prochaines sections.

3. Modèle physique

Le modèle physique a pour but de découpler le modèle logique des modèles du stockage. Ainsi, pour

chaque classe d'entités du modèle logique, un fichier sera associé, et, pour chacun chaque instance d'entité, un enregistrement dans ce fichier. En pratique, pour plusieurs SGBD, ce fichier s'avère plutôt être un groupe de fichiers (métadonnées, données, structures d'index, espaces de débordement).

La mémoire secondaire est structurée en blocs. Une page est composée d'une suite de blocs consécutifs, de façon à permettre d'établir un compromis entre le temps d'accès et le volume d'accès.

En effet, accéder à plusieurs blocs consécutifs requiert généralement moins de ressources que faire une suite d'accès indépendants pour chacun des blocs. Par contre, plus la page est grande, moins la granularité est fine et plus le risque de requérir plus de blocs que nécessaire grandit (et donc de requérir plus de ressources que nécessaire).

Les ressources comprennent principalement le temps (délai, durée), l'énergie, la mémoire (à tous les niveaux) et les canaux (de communication).

La couche physique traduit donc les commandes de la couche logique (exprimée en termes d'opérations sur les classes d'entités du modèle logique) en commandes destinées aux dispositifs de stockage (exprimées en termes de blocs ou de pages). Les opérations offertes par le modèle physique au modèle logique varieront selon ce dernier. Typiquement, pour le modèle relationnel, il comprendra minimalement les opérateurs de l'algèbre relationnelle.

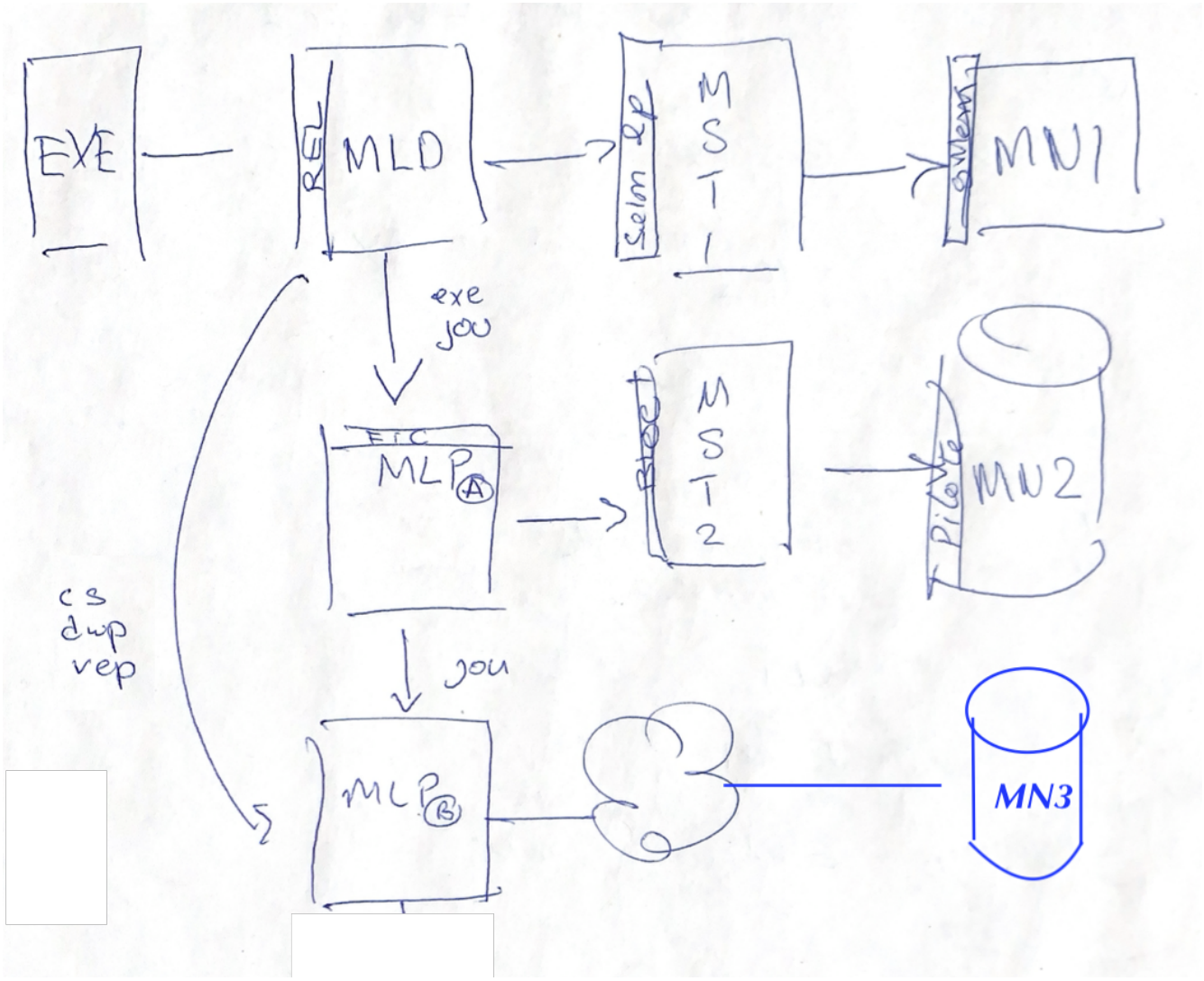

Figure 1. Esquisse d'une architecture simple

Typiquement, le modèle de stockage doit permettre minimalement :

- la définition et allocation d'un fichier sur la base du type d'enregistrement,
- la définition et allocation d'index sur un fichier,
- l'ajout d'un ensemble d'enregistrements à un fichier,
- le retrait d'un ensemble d'enregistrements à un fichier (identifiés par leurs clés),
- le parcours des enregistrements d'un fichier (éventuellement selon un ordre spécifié en termes des clés),
- l'accès à un enregistrement identifié par une de ses clés,
- la modification des attributs d'un enregistrement identifié par une de ses clés.

La mise en oeuvre du modèle de stockage nécessite, notamment pour des raisons d'efficience, le recours à l'indexation (traitée dans le module suivant) et de la pagination (intermédiaire incontournable entre deux niveaux de mémoires).

Dans le modèle physique, les données sont organisées sous la forme d'un ensemble de fichiers. Les fichiers sont eux-mêmes une liste d'enregistrements. Typiquement chaque fichier représente une classe d'entités. Dans un SGBDR, chaque fichier (ou groupe de fichiers) représentera une variable de relation.

3.1. Fichier

Un fichier est une séquence d'enregistrements. Les enregistrements d'un fichier peuvent être de taille fixe (*fixed-length record*) ou variable (*variable-length records*).

3.2. Type d'enregistrement

Un type d'enregistrement (*record type*) est défini par une liste d'attributs (champ, *field*) chacun identifié par un identifiant (nom, *name*), défini par un type associé à un emplacement.

L'adresse et la structure de l'emplacement varie généralement selon que les valeurs sont de taille fixe ou variable.

Typiquement tous les attributs de taille fixe sont stockés dans un enregistrement principal à une adresse déterminée et occupant un espace déterminé. On ajoute ensuite un emplacement de taille fixe pour chacun des attributs de types à taille variable. La structure de ces emplacements varie en fonction du type. En général, ils contiennent la taille effective de la valeur et son adresse dans un espace de débordement prédéterminé. Parfois, ils peuvent contenir la taille effective de la valeur, une signature de la valeur, voire une portion (de taille fixe) de la valeur.

3.3. Enregistrement

Chaque enregistrement (*record*) est formé d'une liste de valeurs. Chaque valeur est formée d'un ou de plusieurs octets qui correspondent à un attribut (du type) de l'enregistrement. Dans un SGBD, les enregistrements correspondent aux instances d'entités. Dans un SGBDR, ils correspondent aux tuplets des variables de relation.

Lorsque la taille des valeurs d'un même attribut varie, une partie fixe est conservée dans l'enregistrement et le complément est relocalisé dans une zone de débordement commune à plusieurs enregistrements.

Dans certaines applications de base de données, il peut s'avérer nécessaire de stocker des éléments de données constitués de grands objets non structurés, qui représentent des images, des vidéos et des sons. On parle alors d'objets binaires de grande taille (*binary large object*, BLOB). Un BLOB est généralement stocké séparément de son enregistrement dans un groupe (pool) de blocs de disque, et un pointeur vers le BLOB est inclus dans l'enregistrement.

En fait, le BLOB est un cas particulier de valeur de taille variable souvent traité de façon distincte des autres valeurs de taille variable en raison de sa très grande taille. La différenciation s'opère généralement en

fonction de la taille moyenne de la valeur :

- moins d'une page, les différents types partagent une même espace de débordement utilisant une même fonction de compression ;
- plus d'une page, chaque type a un espace dédié avec sa propre fonction de compression.

3.4. Unité d'allocation, bloc et page

L'**unité d'allocation physique** (**UAP**, en anglais *physicial record unit* ou *PRU*) est la plus petite unité mémoire adressabble et modifiable du dispositif physique (disque, SSD, etc.).

Pour affranchir le mécanisme de gestion des fichiers de la variabilité de la taille des UAP en fonction des dispositifs de stockage, un **bloc** (représentant un nombre entier d'UAP consécutives) est généralement défini.

De même, une **page**, constituée d'un nombre entier de blocs, est l'unité atomique utilisée par le mécanisme de pagination en mémoire primaire pour gérer les variables de relation, les journaux, les copies de sécurité, etc.

Allocation des fichiers en mémoire secondaire

Il existe plusieurs techniques d'allocation des fichiers en mémoire secondaire (disque, SSD, etc.) :

- Une allocation contigüe (*contiguous allocation*), les blocs sont alloués de façon contigüe. Cette technique rend la lecture globale d'un fichier très rapide, mais la variation de la taille du fichier devient plus difficile.
- Une allocation chainée (*linked allocation*), chaque unité d'allocation contient un pointeur vers la suivante. Un fichier est donc une liste chainée (parfois doublement chainée). Cette technique rend la variation de la taille du fichier plus facile, mais la lecture plus lente.
- Une allocation indexée (*indexed allocation*), une structure d'index est associée au fichier, l'index référant aux unités d'allocation constituant le fichier. Cette technique permet l'accès rapide aux enregistrements du fichier sur la base d'une clé tout en permettant une variation aisée de la taille du fichier. Selon la technique d'indexation utilisée, elle peut aussi permettre un accès séquentiel performant.

Vocabulaire

La dénomination des unités d'allocation (bloc, page, unité) varie d'un auteur à l'autre, d'un domaine à l'autre (architecture des ordinateurs, système d'exploitation, systèmes de bases de données) ; il serait beaucoup trop simple d'avoir un vocabulaire commun, cela gâcherait le plaisir.

Exemple

En PostgreSQL, la taille par défaut d'une page est identique à celle du bloc, soit, par défaut, 8192 octets. Cette valeur est paramétrable au moment du déploiement du SGBD. Elle sera commune à toutes les BD qu'il hébergera.

3.5. Entête d'un fichier

Un entête de fichier contient des informations qui sont nécessaires au programme qui accède aux enregistrements du fichier (les métadonnées du fichier). L'entête comprend des informations permettant de déterminer les adresses de disque des blocs de fichier ainsi la description complète du type des enregistrements.

3.6. Opérations sur les fichiers

La liste des opérations varie beaucoup selon la présence (ou non) d'un index et le cas échéant, le type de

l'index, on retrouve :

- obtention de l'entête d'un fichier,
- obtention d'une unité d'allocation du fichier selon son adresse,
- obtention d'une unité d'allocation du fichier selon une clé d'index,
- parcours d'un fichier,
- parcours de l'index d'un fichier,
- ajout d'une unité d'allocation du fichier selon son adresse,
- retrait d'une unité d'allocation du fichier selon son adresse,
- modification d'une unité d'allocation du fichier selon son adresse,
- etc.

3.7. Catalogue

La liste des fonctionnalités varie beaucoup selon la structure du catalogue (en particulier selon qu'il est hiérarchisé ou non) minimalement, on retrouve :

- obtention d'une liste de fichiers selon certains critères,
- ajout d'un fichier,
- retrait d'un fichier,
- ajout d'un dossier dans le catalogue,
- retrait d'un dossier dans le catalogue,
- etc.

3.7.1. Hiérarchisation (ou non) du catalogue

La hiérarchisation du catalogue (par l'entremise de dossiers, sous-dossiers, etc.) n'est pas requise par le stockage des données du SGBD. Elle peut cependant être prise en charge, notamment lorsque le système de gestion de fichiers du système d'exploitation sous-jacent est utilisé. Lorsqu'un système propre au SGBD est privilégié, elle est souvent omise.

3.7.2. Indirection (ou non) des fichiers

La double indirection au niveau du catalogue des fichiers est un impératif de performance. L'absence de redirection est souvent un motif pour lequel les systèmes de fichiers de certains systèmes d'exploitation ne seront pas retenus.

Voici quelques exemples :

- Poursuite du traitement sans perturbation en cas de renommage.
- Poursuite du traitement en cas de relocalisation.
- Préparation d'une mise à jour d'un fichier en concurrence avec l'exploitation de la valeur précédente du fichier. Au moment opportun (déterminé par le gestionnaire transactionnel), le deuxième pointeur est simplement modifié, sans impact sur les références conservées en mémoire primaire.

3.8. Organisation des dispositifs matériels

Bien qu'un seul dispositif matériel puisse être utilisé (voir un petit nombre), le plus souvent, plusieurs seront associés afin de satisfaire aux critères de cout, de capacité ou de délai. L'utilisation de plusieurs dispositifs est aussi un élément de palliation déterminant.

- NAS

- SAN
- Miroir
- RAID
- autres

En direct, en présence, au tableau !

4. Pagination

En direct, en présence, au tableau !

Pour informations complémentaires, voir

- [Lelarge2023a], chapitres 4 et 5
- [Silberschatz2021a], chapitres 9 et 10
- [Tanenbaum2015a], chapitres 3

Références

[Elmasri2016]
Ramez ELMASRI et Shamkant B. NAVATHE;
Fundamentals of database systems;
7th Edition, Pearson, Hoboken (NJ, US), 2016;
ISBN 978-0-13-397077-7.

[Hainaut2022]
Jean-Luc HAINAUT;
Bases de données - concepts, utilisation et développement;
5e édition, Dunod, 2022;
ISBN 978-2-10-084285-8.

[Lelarge2023a]
Guillaume LELARGE, Juilen ROUOHAUD;
PostgreSQL : architecture et notions avancées;
5e édition, Éditions D_Booker, 2023;
ISBN 978-2-8227-1124-1.

[Petrov2019]
Alex PETROV;
Datavase internals - A deep dive into how distributed data systems work;
1re édition, 3e révision, O'Reilly, 2020;
ISBN 978-1-492-04034-7.

[Sciore2020a]
Edward SCIORE;
Database design and implementation;
Second edition, Springer, 2020;
ISBN 978-3-030-33855-0.

[Silberschatz2018]
Abraham SILBERSCHATZ, Peter B. GALVIN, Greg GAGNE;
Operating System Concepts;
10th Edition, Wiley, 2018;
ISBN 978-1119800361.

[Tanenbaum2015a]
Andrew S. TANENBAUM, Herbert BOS;
Modern Operating System;
4th Edition, Pearson, 2015;
ISBN 978-0-13-359162-0.

Produit le 2025-10-01 10:24:29 UTC

Université de Sherbrooke