A faded comic book illustration serves as the background. It depicts a man with a pipe and a woman in a red dress. The man is in the foreground, looking towards the right. The woman is in the background, looking towards the man. The overall style is reminiscent of mid-20th-century comic art.

SQL Server VLDB  
very **large** databases

Arian PAPILLON

#datafrogs

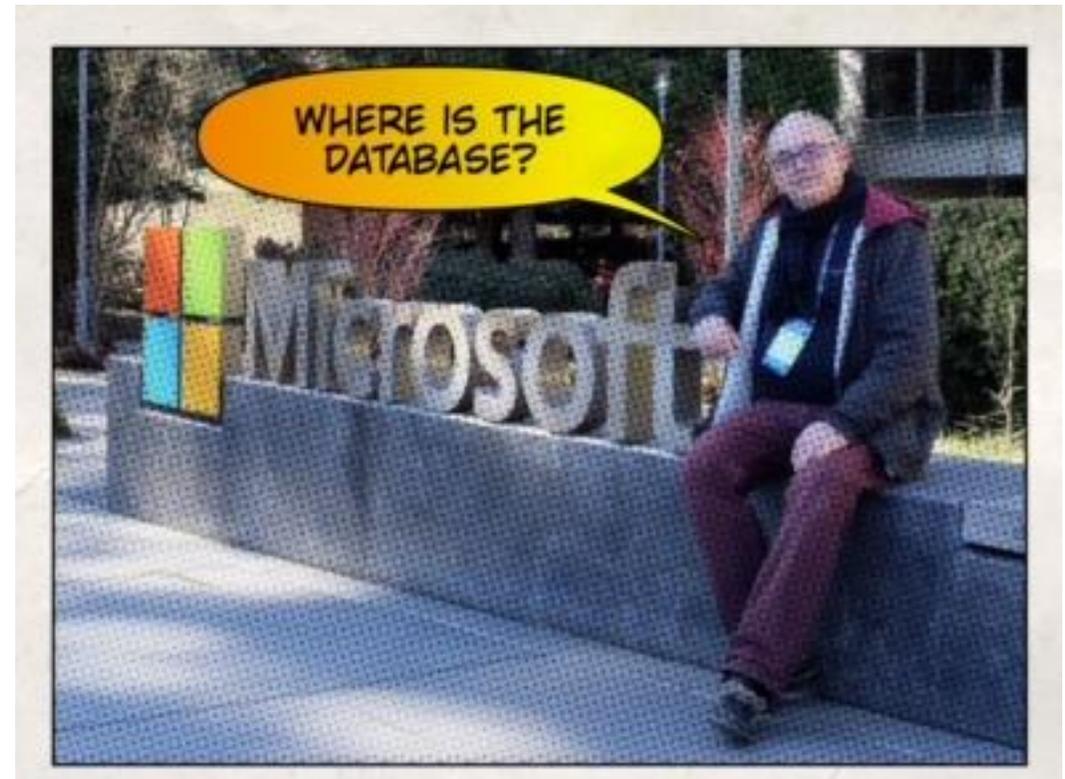
# Qui suis-je ?



- **Arian PAPILLON**

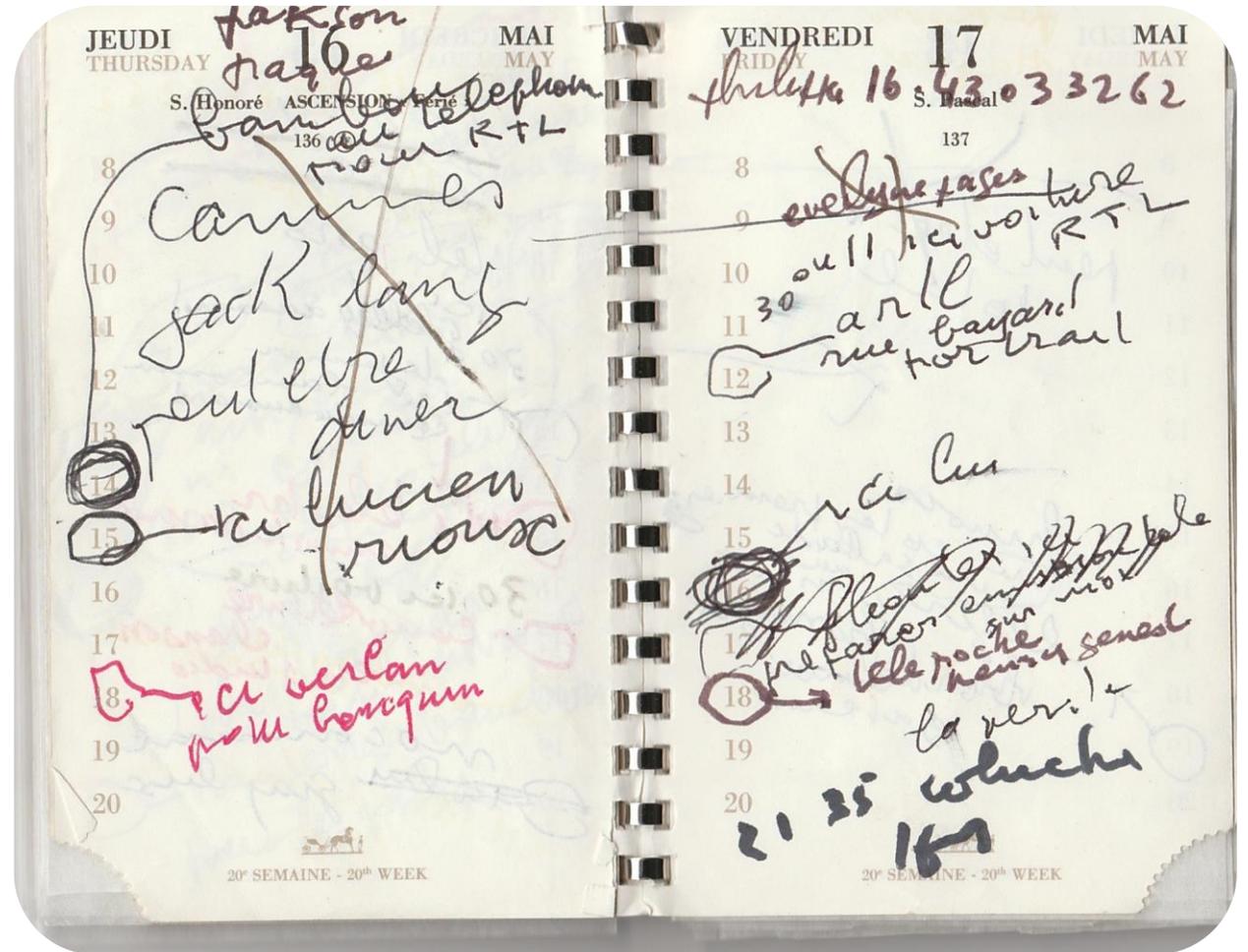


- Datafly (depuis 2008)
- Expert spécialiste bases de données (SQL Server et BI)
- Quelques années d'expériences (25 ans)
- MVP Data Platform (10 ans)
- <http://www.datafly.fr> & <http://blog.datafly.fr>
- <http://www.mssql.fr>
- <https://youtube.com/datafly>
- <https://github.com/datafly/ssmsinforeports>
- <https://www.linkedin.com/in/arianpapillon/>



# Agenda

- SGBD ou big data ?
- Le stockage
- Le partitionnement
- Les vues partitionnées distribuées
- L'administration et la maintenance
- L'optimisation



# Sondage

- Sondage : quelles est la taille de votre plus grande base de données SQL Server ?
  - < 100 GB
  - < 500 GB
  - < 1 TB
  - 1 - 5 TB
  - 5 - 10 TB
  - 10 à 20 TB
  - 20 à 50 TB
  - > 50 TB

<https://j.mp/3tsxFba>

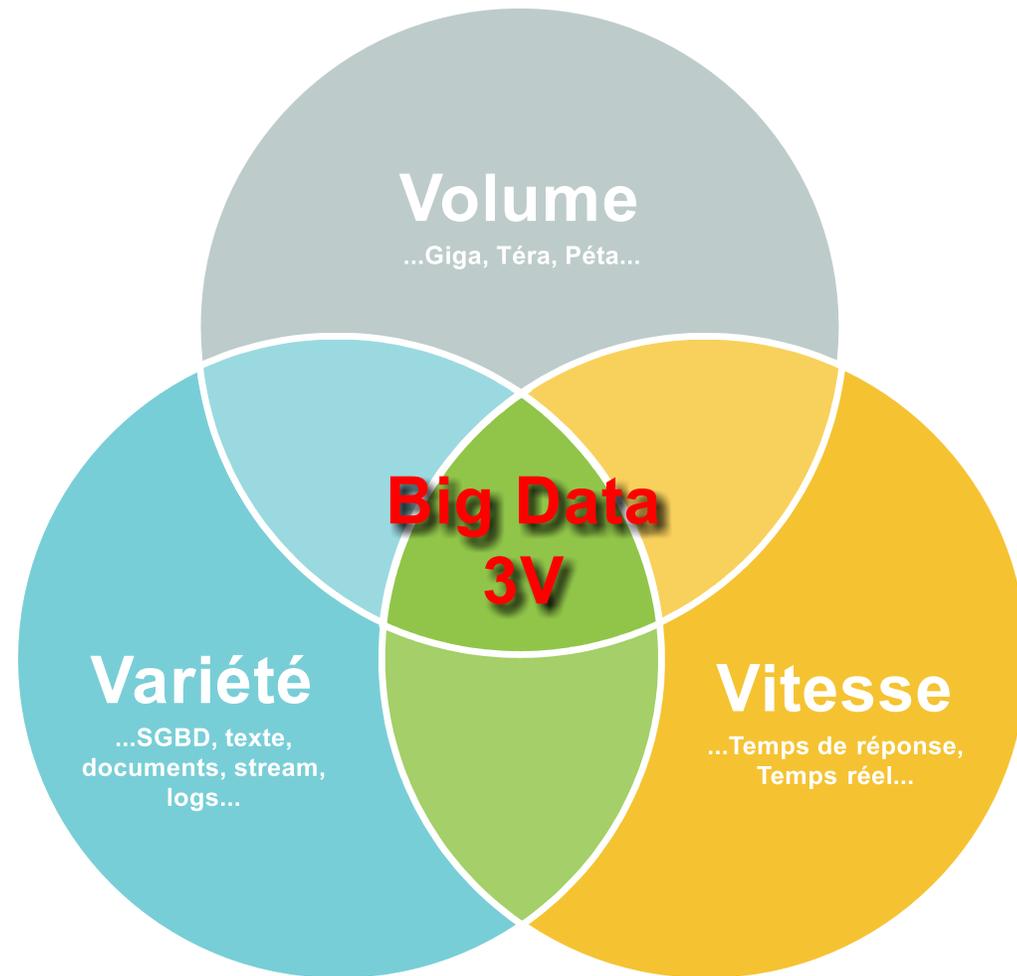


# Very large comment ?

- Capacités maximales de SQL Server
  - Taille maximum d'une base : 524272 To (512 Po)
  - Nombre maximum de groupes de fichiers : 32767
  - Taille maximum d'un fichier de données : 16 To
  - Taille maximum d'un fichier journal : 2 To
- (Very ?) large si la volumétrie se compte en Terabytes...



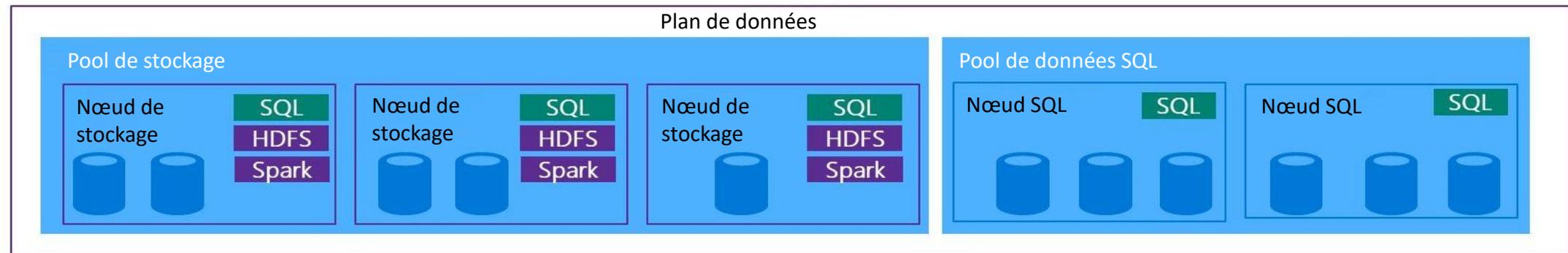
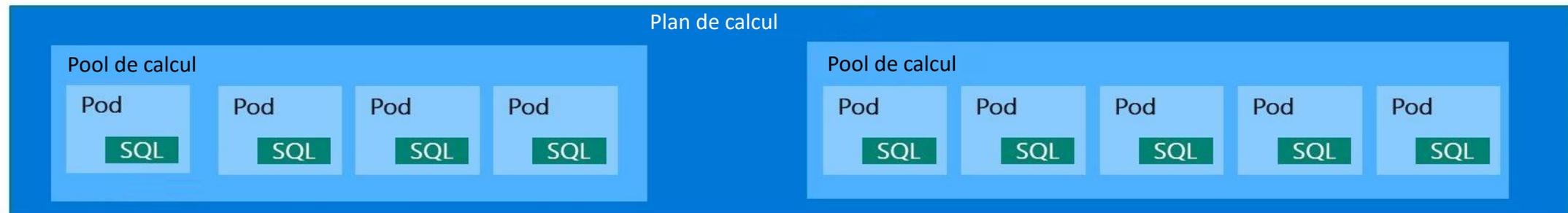
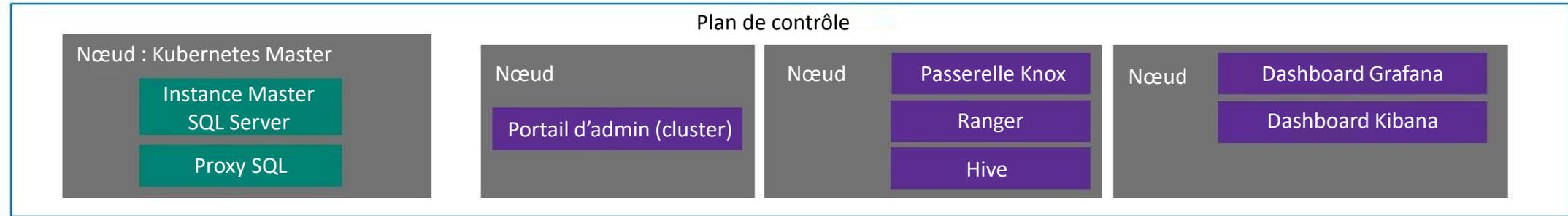
# SGBD ou Big Data



- Architecture big data si :
  - Stockage de données non structurées
  - Volumétries  $\geq 10$  TB+ avec évolution forte
    - Stockage distribué
    - Calcul distribué



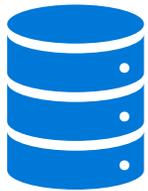
# Cluster SQL Server Big Data



Mais revenons au bon vieux SGBD



# Conception : bonnes pratiques



## Modèle

Une base de données bien modélisée tient moins de place



## Types de données

Le choix des bons types de données est essentiel



## Indexation

Une clé d'index clustered doit être petite, ...

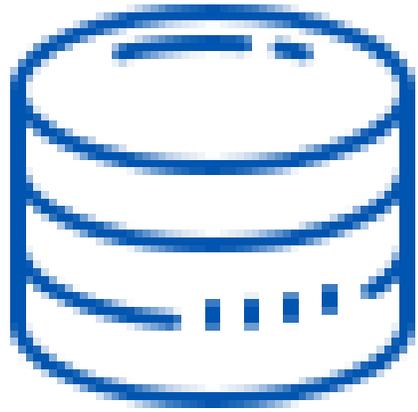
# Stockage

## Encore des bonnes pratiques

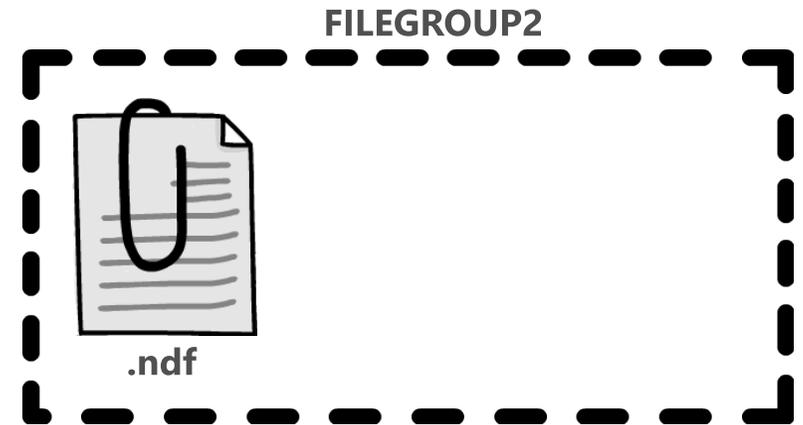
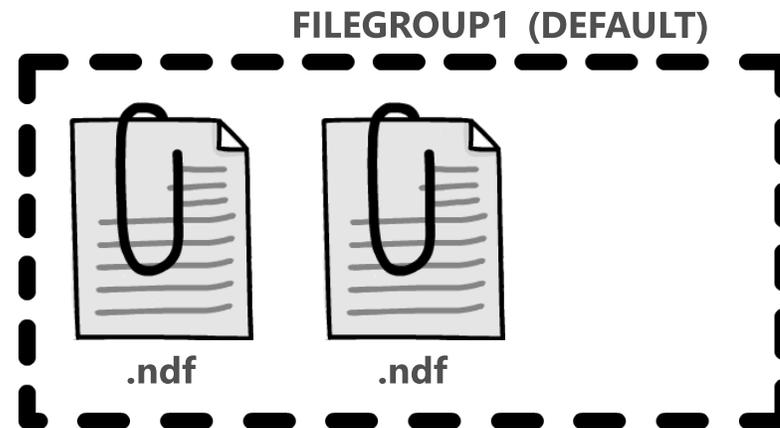
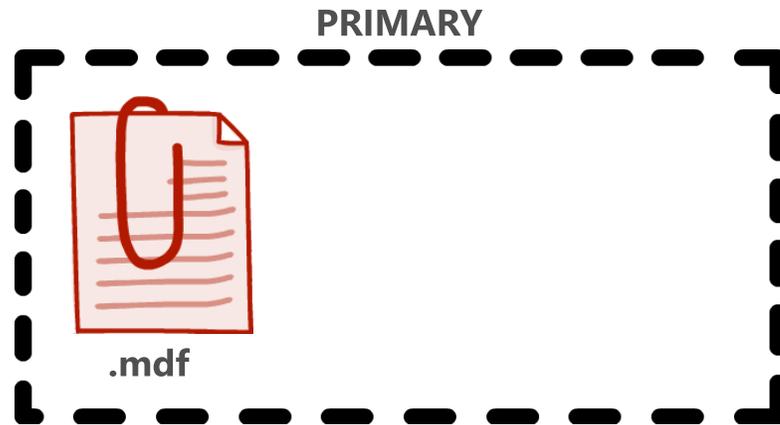
- Pré-dimensionner les fichiers
- Tailler les incréments de croissance
- Fuir les shrink inutiles
- Activer l'initialisation instantanée
- Performances du stockage
  - RAID, etc...



# Rappel : stockage d'une base de données



.ldf



```
CREATE TABLE ...  
[ ON {filegroup | partition_scheme (...)}  
]
```



# Diviser le stockage : files et filegroups

- A penser (si possible) dès la création de la base !
- Pour répartir la base sur plusieurs disques
  - Pour améliorer les performances
- Pour manipuler des fichiers de taille raisonnable
- Pour faciliter la maintenance
  - CHECKDB -> CHECKFILEGROUP
- Pour permettre des sauvegardes et restaurations partielles
  - BACKUP DATABASE [MyDB] FILEGROUP = [MyFG]



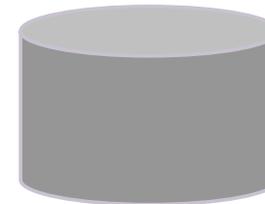
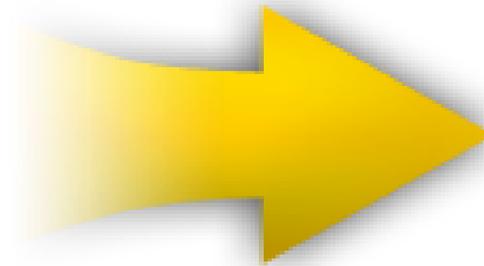
# Diviser le stockage : partitionnement

- A concevoir (si possible) dès la conception de la base
  - Pour les très grosses tables
  - Pour dépasser les limites de taille de mes volumes disque
  - Pour améliorer les performances
    - Pas forcément...
  - Pour gérer la purge, l'import ou l'archivage
  - Pour passer les données froides en lecture seule
    - Et ne plus sauvegarder que la partie active de mes tables



# Le principe du partitionnement

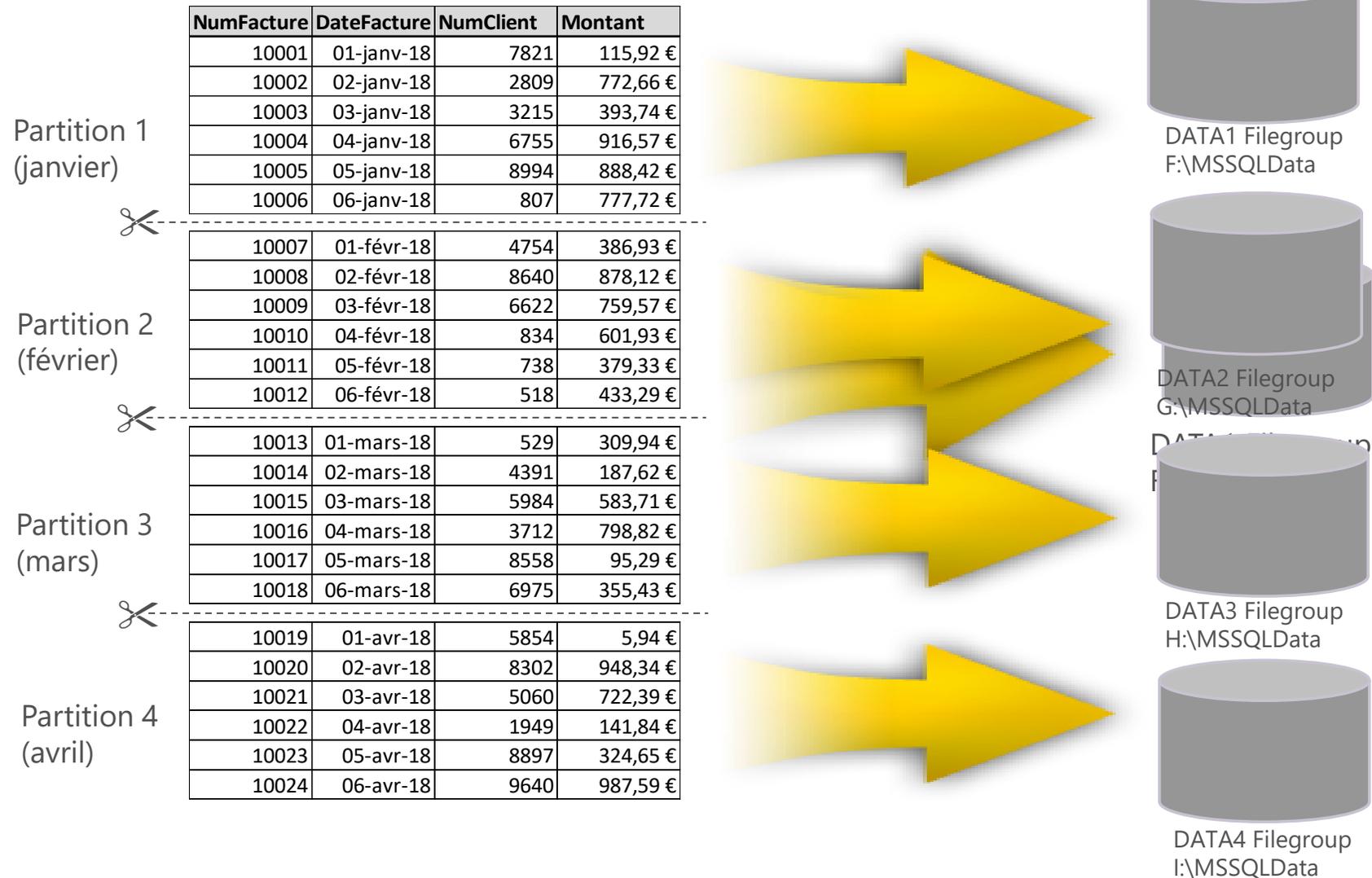
NumFacture	DateFacture	NumClient	Montant
10001	01-janv-18	7821	115,92 €
10002	02-janv-18	2809	772,66 €
10003	03-janv-18	3215	393,74 €
10004	04-janv-18	6755	916,57 €
10005	05-janv-18	8994	888,42 €
10006	06-janv-18	807	777,72 €
10007	01-févr-18	4754	386,93 €
10008	02-févr-18	8640	878,12 €
10009	03-févr-18	6622	759,57 €
10010	04-févr-18	834	601,93 €
10011	05-févr-18	738	379,33 €
10012	06-févr-18	518	433,29 €
10013	01-mars-18	529	309,94 €
10014	02-mars-18	4391	187,62 €
10015	03-mars-18	5984	583,71 €
10016	04-mars-18	3712	798,82 €
10017	05-mars-18	8558	95,29 €
10018	06-mars-18	6975	355,43 €
10019	01-avr-18	5854	5,94 €
10020	02-avr-18	8302	948,34 €
10021	03-avr-18	5060	722,39 €
10022	04-avr-18	1949	141,84 €
10023	05-avr-18	8897	324,65 €
10024	06-avr-18	9640	987,59 €



DATA1 Filegroup  
F:\MSSQLData



# Le principe du partitionnement



# Manipuler les partitions : purge de données

Table Facture

NumFacture	DateFacture	NumClient	Montant
10001	01-janv-18	7821	115,92 €
10002	02-janv-18	2809	772,66 €
10003	03-janv-18	3215	393,74 €
10004	04-janv-18	6755	916,57 €
10005	05-janv-18	8994	888,42 €
10006	06-janv-18	807	777,72 €
✂			
10007	01-févr-18	4754	386,93 €
10008	02-févr-18	8640	878,12 €
10009	03-févr-18	6622	759,57 €
10010	04-févr-18	834	601,93 €
10011	05-févr-18	738	379,33 €
10012	06-févr-18	518	433,29 €
✂			
10013	01-mars-18	529	309,94 €
10014	02-mars-18	4391	187,62 €
10015	03-mars-18	5984	583,71 €
10016	04-mars-18	3712	798,82 €
10017	05-mars-18	8558	95,29 €
10018	06-mars-18	6975	355,43 €
✂			
10019	01-avr-18	5854	5,94 €
10020	02-avr-18	8302	948,34 €
10021	03-avr-18	5060	722,39 €
10022	04-avr-18	1949	141,84 €
10023	05-avr-18	8897	324,65 €
10024	06-avr-18	9640	987,59 €

Partition 1  
(janvier)



Partition 2  
(février)



Partition 3  
(mars)



Partition 4  
(avril)

Switch

Table PurgeFacture

NumFacture	DateFacture	NumClient	Montant
10001	01-janv-18	7821	115,92 €
10002	02-janv-18	2809	772,66 €
10003	03-janv-18	3215	393,74 €
10004	04-janv-18	6755	916,57 €
10005	05-janv-18	8994	888,42 €
10006	06-janv-18	807	777,72 €



# Manipuler les partitions : import de données

Table AnciennesFactures

NumFacture	DateFacture	NumClient	Montant
10001	01-janv-17	6767	452,24 €
10002	02-janv-17	8941	246,15 €
10003	03-janv-17	9414	324,78 €
10004	04-janv-17	9979	148,66 €
10005	05-janv-17	8085	317,41 €
10006	06-janv-17	6620	63,26 €

Switch

Partition 1

Table Facture

NumFacture	DateFacture	NumClient	Montant
10001	01-janv-17	6767	452,24 €
10002	02-janv-17	8941	246,15 €
10003	03-janv-17	9414	324,78 €
10004	04-janv-17	9979	148,66 €
10005	05-janv-17	8085	317,41 €
10006	06-janv-17	6620	63,26 €



Partition 2  
(janvier)

10001	01-janv-18	7821	115,92 €
10002	02-janv-18	2809	772,66 €
10003	03-janv-18	3215	393,74 €
10004	04-janv-18	6755	916,57 €
10005	05-janv-18	8994	888,42 €
10006	06-janv-18	807	777,72 €



Partition 3  
(février)

10007	01-févr-18	4754	386,93 €
10008	02-févr-18	8640	878,12 €
10009	03-févr-18	6622	759,57 €
10010	04-févr-18	834	601,93 €
10011	05-févr-18	738	379,33 €
10012	06-févr-18	518	433,29 €



Partition 4  
(mars)

10013	01-mars-18	529	309,94 €
10014	02-mars-18	4391	187,62 €
10015	03-mars-18	5984	583,71 €
10016	04-mars-18	3712	798,82 €
10017	05-mars-18	8558	95,29 €
10018	06-mars-18	6975	355,43 €



# Manipuler les partitions : archivage de données

Table Facture

NumFacture	DateFacture	NumClient	Montant
10001	01-janv-18	7821	115,92 €
10002	02-janv-18	2809	772,66 €
10003	03-janv-18	3215	393,74 €
10004	04-janv-18	6755	916,57 €
10005	05-janv-18	8994	888,42 €
10006	06-janv-18	807	777,72 €
10007	01-févr-18	4754	386,93 €
10008	02-févr-18	8640	878,12 €
10009	03-févr-18	6622	759,57 €
10010	04-févr-18	834	601,93 €
10011	05-févr-18	738	379,33 €
10012	06-févr-18	518	433,29 €
10013	01-mars-18	529	309,94 €
10014	02-mars-18	4391	187,62 €
10015	03-mars-18	5984	583,71 €
10016	04-mars-18	3712	798,82 €
10017	05-mars-18	8558	95,29 €
10018	06-mars-18	6975	355,43 €
10019	01-avr-18	5854	5,94 €
10020	02-avr-18	8302	948,34 €
10021	03-avr-18	5060	722,39 €
10022	04-avr-18	1949	141,84 €
10023	05-avr-18	8897	324,65 €
10024	06-avr-18	9640	987,59 €

Partition 1  
(janvier)



Partition 2  
(février)



Partition 3  
(mars)



Partition 4  
(avril)

Switch

Switch

Switch

Table ArchiveFacture

NumFacture	DateFacture	NumClient	Montant
10001	01-janv-18	7821	115,92 €
10002	02-janv-18	2809	772,66 €
10003	03-janv-18	3215	393,74 €
10004	04-janv-18	6755	916,57 €
10005	05-janv-18	8994	888,42 €
10006	06-janv-18	807	777,72 €
10007	01-févr-18	4754	386,93 €
10008	02-févr-18	8640	878,12 €
10009	03-févr-18	6622	759,57 €
10010	04-févr-18	834	601,93 €
10011	05-févr-18	738	379,33 €
10012	06-févr-18	518	433,29 €
10013	01-mars-18	529	309,94 €
10014	02-mars-18	4391	187,62 €
10015	03-mars-18	5984	583,71 €
10016	04-mars-18	3712	798,82 €
10017	05-mars-18	8558	95,29 €
10018	06-mars-18	6975	355,43 €

Partition 1  
(janvier)



Partition 2  
(février)



Partition 3  
(mars)



# Partitionner pour réduire le volume des sauvegardes

Table Facture

	NumFacture	DateFacture	NumClient	Montant	
Partition 1 (janvier)	10001	01-janv-18	7821	115,92 €	ReadOnly
	10002	02-janv-18	2809	772,66 €	
	10003	03-janv-18	3215	393,74 €	
	10004	04-janv-18	6755	916,57 €	
	10005	05-janv-18	8994	888,42 €	
	10006	06-janv-18	807	777,72 €	
✂					
Partition 2 (février)	10007	01-févr-18	4754	386,93 €	ReadOnly
	10008	02-févr-18	8640	878,12 €	
	10009	03-févr-18	6622	759,57 €	
	10010	04-févr-18	834	601,93 €	
	10011	05-févr-18	738	379,33 €	
	10012	06-févr-18	518	433,29 €	
✂					
Partition 3 (mars)	10013	01-mars-18	529	309,94 €	ReadOnly
	10014	02-mars-18	4391	187,62 €	
	10015	03-mars-18	5984	583,71 €	
	10016	04-mars-18	3712	798,82 €	
	10017	05-mars-18	8558	95,29 €	
	10018	06-mars-18	6975	355,43 €	
✂					
Partition 4 (avril)	10019	01-avr-18	5854	5,94 €	
	10020	02-avr-18	8302	948,34 €	
	10021	03-avr-18	5060	722,39 €	
	10022	04-avr-18	1949	141,84 €	
	10023	05-avr-18	8897	324,65 €	
	10024	06-avr-18	9640	987,59 €	



# Comment partitionner

- Choix de la colonne de partitionnement
- Fonction de partition
- Schéma de partition
- Création des objets partitionnés



# Partitionner les index

- Le partitionnement de l'index CLUSTERED partitionne la table !
- Pour tout index CLUSTERED, sa clé doit contenir la colonne de partition
  - Si l'index CLUSTERED est unique, il faut l'ajouter explicitement,
  - Dans les autres cas SQL Server l'ajoutera de lui-même.
- Les autres index (NONCLUSTERED) peuvent être partitionnés ou non
- Ils peuvent aussi être partitionnés à l'identique de la table : index alignés
  - SQL Server va inclure implicitement la colonne de partitionnement dans l'index (si l'index ne l'inclut pas déjà).
  - Pour les index UNIQUE la colonne de partition doit être explicitement contenue dans la clé d'index



# Gestion des partitions

- Switcher des partitions
- Fusionner des partitions
- Tronquer des partitions
- Ajouter des partitions



# Backups

- Backups par fichiers, par groupes de fichiers
  - Et backups différentiels
- Sauvegarde et restauration partielle, restauration online



# Compression

- Compression de tables et d'index (pour une table/index ou une partition)
  - Niveau ligne
  - Niveau page
- Clustered column store index



# Diviser le stockage : vues partitionnées distribuées

- Partitionner/distribuer les données d'une table sur plusieurs tables
  - Situées dans plusieurs bases
  - ou plusieurs serveurs...!
- Créer une table sur chaque
  - Avec une contrainte CHECK sur la clé de partitionnement
- Créer une vue qui met en UNION ALL toutes les tables



# Vue distribuées : inconvénients

- Lecture dans plusieurs tables si la colonne de partition n'est pas fournie
- Du code pour les insertions, suppressions, mises à jour :
  - Triggers INSTEAD OF pour rediriger les mises à jour
  - Transactions distribuées et DTC
- Gestion des identity et intégrité référentielle
  - Séquences
  - Code
- Eviter les jointures entre serveurs
  - Répliquer les tables jointes pour éviter les jointures inter-serveurs



# Administration

- CHECKDB : consommation de tempdb
  - DBCC CHECKDB () WITH ESTIMATEONLY
  - CHECKFILEGROUP, CHECKTABLE
- Maintenance des index
  - Rebuild par partition
- Maintenance des statistiques
  - Statistiques incrémentales



# Statistiques

- Envisagez la mise à jour auto en mode asynchrone (ou faites vos mises à jour manuellement)

```
ALTER DATABASE [MyVLDB] SET AUTO_UPDATE_STATISTICS_ASYNC
```

```
ALTER DATABASE [MyVLDB] SET AUTO_UPDATE_STATISTICS OFF
```

Pour les tables partitionnées :

- Pensez à activer la création auto de statistiques incrémentales

```
ALTER DATABASE MyVLDB
```

```
SET AUTO_CREATE_STATISTICS ON (INCREMENTAL = ON)
```

- Créez vos nouveaux index partitionnés avec l'option

```
WITH (STATISTICS_INCREMENTAL = ON)
```

- Pour les statistiques ou index déjà existants sur des tables partitionnées, les reconstruire

```
UPDATE STATISTICS MyTable
```

```
(MyStat) WITH FULLSCAN, INCREMENTAL=ON
```

- Ou rebuild d'index avec STATISTICS\_INCREMENTAL=ON

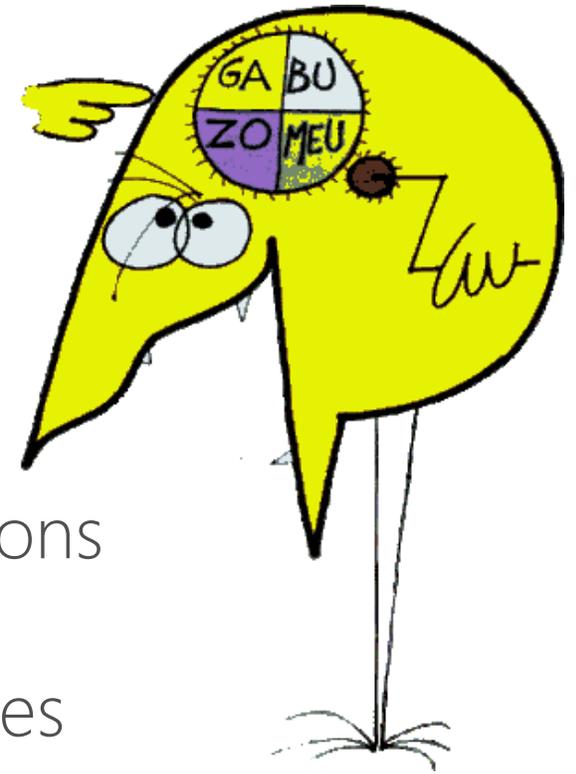


# Optimisation



# Optimisation

- Tailler le hardware en conséquence
- Stockage : taillez large et ayez de l'espace libre
  - Données et journal de transactions : pour les opérations massives
  - Tempdb : pour supporter des tris et objets temporaires volumineux
- Plans de requête : la chasse aux scans
- Eviter les escalades de verrous
- Le mode batch (on column store, on row store)
- Indexation : index couvrants, index columnstore





3E  
C'EST TOUT  
POUR AUJOUR  
D'HUI

#datafrogs





<http://www.datafly.fr>

