Plan d'exécution et opérateurs

Rappel : un **plan d'exécution** est un arbre constitué **d'opérateurs** échangeant des **flux de données**.

Plan d'exécution et opérateurs

Rappel : un **plan d'exécution** est un arbre constitué **d'opérateurs** échangeant des **flux de données**.

Caractéristiques des opérateurs

- ont une forme générique (**itérateurs**) ;
- fournissent une tâche spécialisée (cf. l'algèbre relationnelle)
- peuvent être ou non **bloquants**.

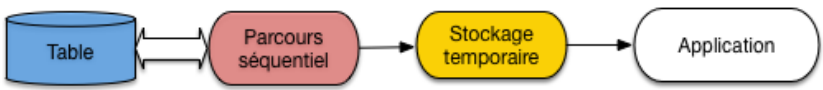
Un petit nombre suffit pour couvrir SQL !

Dans ce cours

On va apprendre à implanter un moteur d'exécution pour (presque toutes) les requêtes SQL !

Mode naïf : matérialisation

Dans ce mode, un opérateur calcule son résultat, puis le transmet.

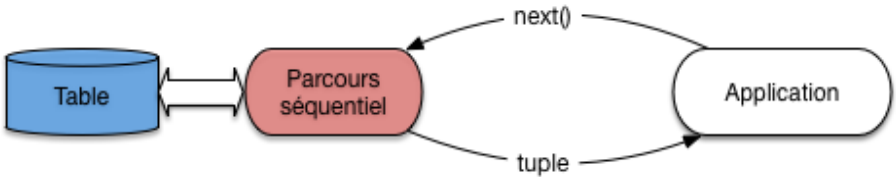


Deux inconvénients :

- Consomme de la mémoire.
- Introduit un temps de **latence**.

La bonne solution : pipelinage

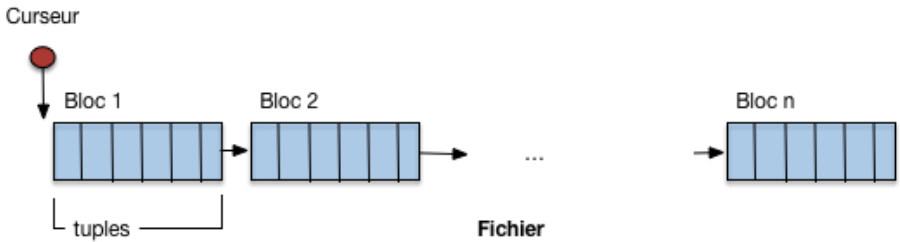
Le résultat est produit **à la demande**.



- **Plus de stockage intermédiaire**.
- **Latence minimale**.

Illustration : l'opérateur `FullScan`

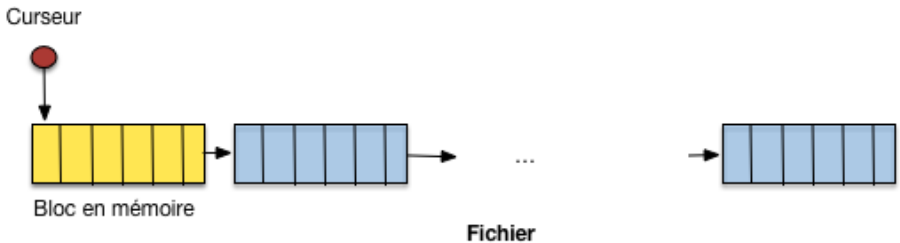
Au moment du `open()`, le curseur est positionné **avant** le premier nuplet.



`open()` désigne la phase d'initialisation de l'opérateur.

Illustration : l'opérateur `FullScan`

Le premier `next()` entraîne l'accès au premier bloc, placé en mémoire.



Le curseur se place sur le premier nuplet, qui est retourné comme résultat. **Le temps de réponse est minimal**.

Illustration : l'opérateur `FullScan`

Le deuxième `next()` avance d'un cran dans le parcours du bloc.

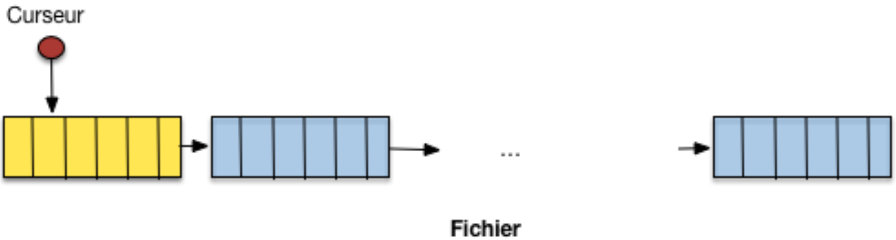


Illustration : l'opérateur `FullScan`

Après plusieurs `next()`, le curseur est positionné sur le dernier nuplet du bloc.

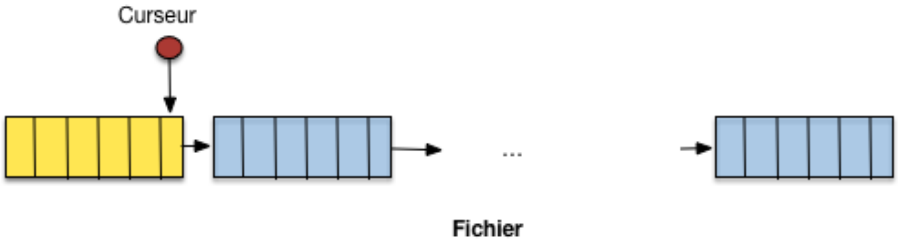
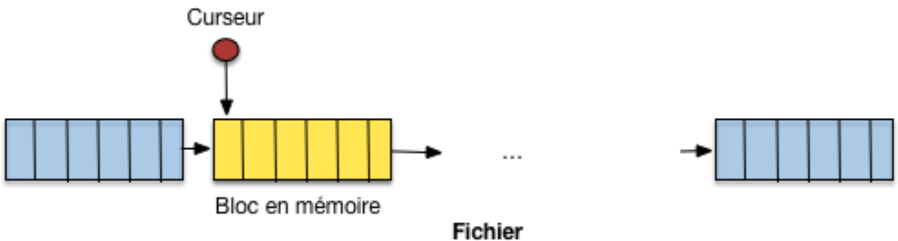


Illustration : l'opérateur `FullScan`

L'appel suivant à `next()` charge le second bloc en mémoire.



Bilan : besoin en mémoire réduit (1 bloc) ; temps de réponse très court.

Opérateur bloquant

Tous les opérateurs peuvent-ils fonctionner en mode pipelinage ?

```
select min(date) from T
```

On ne peut pas produire un nuplet avant d'avoir examiné **toute** la table.

Il faut alors introduire un opérateur **bloquant**, avec une latence forte.

Avec un opérateur bloquant, on **additionne** le temps d'exécution et le temps de traitement.

Résumé : opérateurs d'exécution

Principes essentiels :

1. **Itération** : dans tous les cas, un opérateur produit les nuplets à la demande.
2. **Pipelinage** : si possible, le résultat est calculé au fur et à mesure.
3. **Matérialisation** : parfois le résultat intermédiaire doit être calculé et stocké.

Bien distinguer

- **Temps de réponse** : temps pour obtenir le premier nuplet.
- **Temps d'exécution** : temps pour obtenir tous les nuplets.

Résumé : opérateurs d'exécution

Principes essentiels :

1. **Itération** : dans tous les cas, un opérateur produit les nuplets à la demande.
2. **Pipelinage** : si possible, le résultat est calculé au fur et à mesure.
3. **Matérialisation** : parfois le résultat intermédiaire doit être calculé et stocké.

Bien distinguer

- **Temps de réponse** : temps pour obtenir le premier nuplet.
- **Temps d'exécution** : temps pour obtenir tous les nuplets.

Merci !