

A large, stylized, light-colored dog face is centered in the background. It has two large, round, grey eyes with white highlights, a small grey nose, and a simple grey mouth. The face is set against a light yellow background with two purple, teardrop-shaped shapes in the upper corners.

パフォーマンスチューニングの第一歩 [DBの動きを知ろう]

夏椰(今川 美保)

何を考える？

- SQLを書くときに何を考えている？
 - 欲しいデータを取得する条件？
 - どのテーブルをつなげる？

...etc.

データを取得すると・・・

- バッファというメモリのデータを探す。
- ディスクからデータが読み取られる。
- 読み取ったデータをバッファに格納する。
- バッファに出したデータを選別する。
- データの計算を行う。

→バッファがいっぱいになったら？

古いデータを消しちゃう！

ディスクから取得する

- ディスクアクセスする回数は少ないほうがいい。
 - ディスクへアクセスする時間も短いほうがいい。
- ディスクアクセスする単位は？

テーブルデータの格納

- ページ

- 1ページ = 8,192B

- エクステント

- エクステント = 8ページ
= 64KB

1エクステント

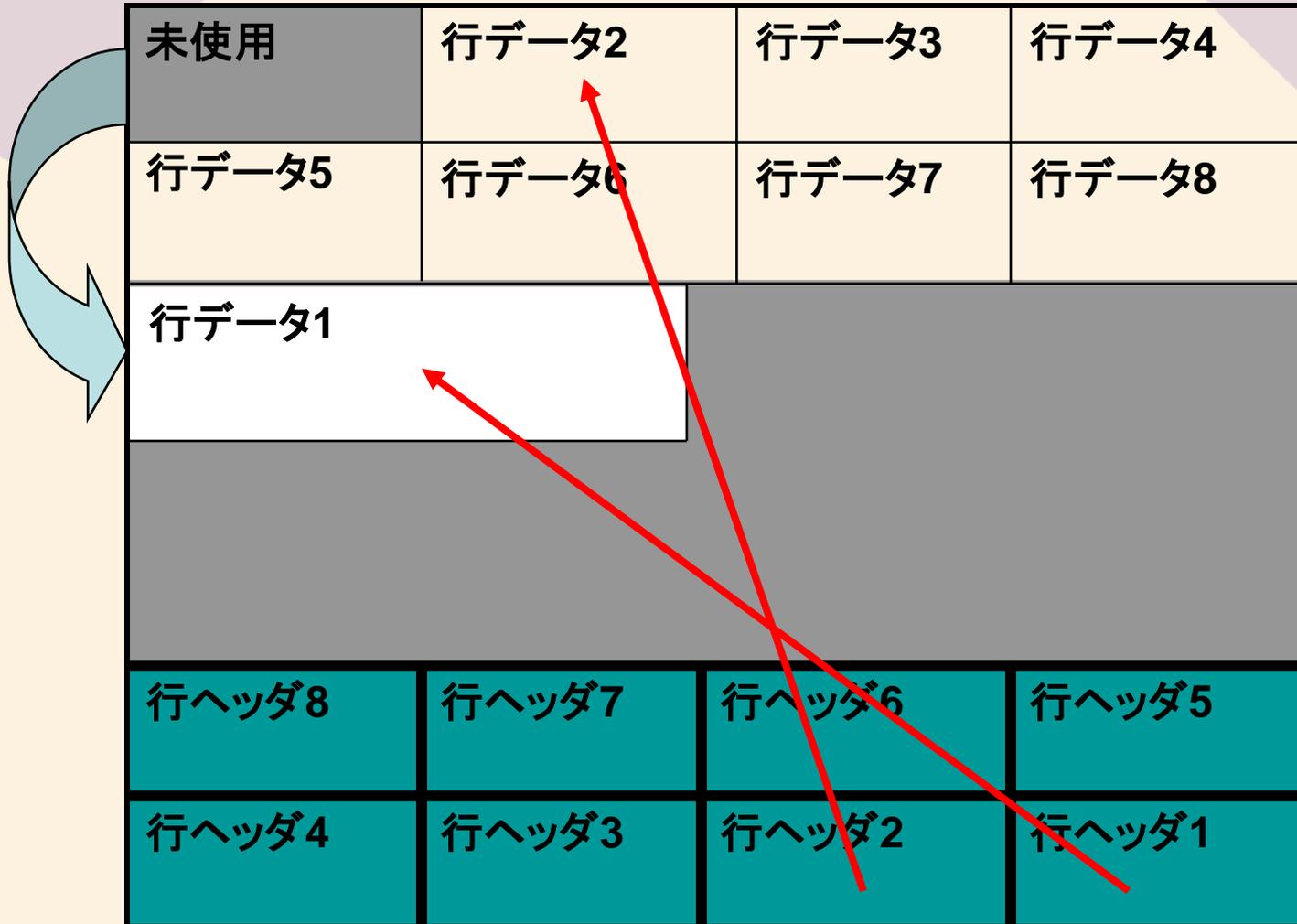
1ページ							
------	--	--	--	--	--	--	--

行の格納

行データ1	行データ2	行データ3	行データ4
行データ5	行データ6	行データ7	行データ8
未使用			
行ヘッダ8	行ヘッダ7	行ヘッダ6	行ヘッダ5
行ヘッダ4	行ヘッダ3	行ヘッダ2	行ヘッダ1

The diagram illustrates row storage in a table. It shows a 4x4 grid of cells. The top two rows contain data (行データ1-8). The third row is shaded gray and labeled '未使用' (Unused). The bottom two rows contain headers (行ヘッダ1-8). Red arrows indicate that data from row 2 (行データ5 and 6) is being moved to row 1 (行データ1 and 2), and data from row 1 (行データ3 and 4) is being moved to row 2 (行データ7 and 8). This suggests a row-major storage order where data is stored row by row, but headers are stored in reverse order.

行の格納



簡単に考えると

- ・ ディスクアクセスする回数は少ないほうがいい。
 - 1ページに格納されている行数が多いといいね。
 - 1ページ内の断片化が少ないといいね。
 - 1ページ内に欲しいデータがたくさんあるといいね。

テーブルの種類

・テーブルのタイプ

- ・ テーブル・インデクス分離型
 - ヒープ(非クラスタテーブル)
- ・ テーブル・インデクス一体型
 - クラスタテーブル

ヒープ

- データは入れた順に格納されていく。
- 格納されたデータが移動するのは
UPDATEによってデータ長が変更されたとき。
→NULL項目に値を入れた
→データ長自体を変えた。
(VARCHARの格納データ長が増えた等)
- DELETEでも未使用領域が発生する。

クラスタ化テーブル

- データは入れた順に格納されていく。
- 格納されたデータが移動するのは
UPDATEによってデータ長が変更されたとき。
→NULL項目に値を入れた
→データ長自体を変えた。
(VARCHARの格納データ長が増えた等)
- DELETEでも未使用領域が発生する。
- **クラスタキーの更新**

この場合どっちがいい？

さてこの場合、ヒープにしますか？

クラスタ化テーブルにしますか？

注文番号	顧客コード	金額	時間
0001	K0001	1,000	1/22 12:00:05

もし、顧客毎にデータを表示するなら

- 顧客コードにクラスタキーを設定する。
→顧客ごとにデータを固める
- 注文番号にユニークインデクスを作成する。
→注文番号重複を防ぐ

注文番号	顧客コード	金額	時間
0001	K0001	1,000	1/22 12:00:05

まとめ

- 内部動作を理解して 設計するだけで
SQL動作の差が出ます！

内部動作に興味を持ち
よりよい設計にお役立てください。