SQL実践入門Night
~SQLの深かったり深くなかったりする話~

2018.01.24 @ 株式会社VOYAGE GROUP

講演者：ミック

自己紹介

- 退役DBエンジニア。BI/DWHのシステム開発を経験したのち、パフォーマンス専門チームで性能設計やチューニングを数年間担当。現在は主に技術者育成と社内政治に従事。

- 本日のHash Tag #sqljissen

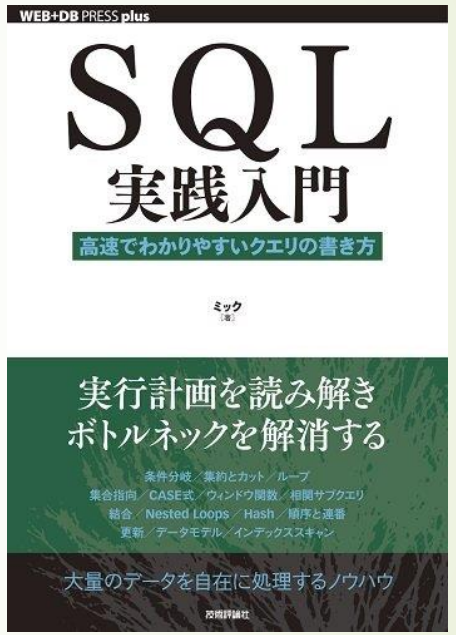
本日のアジェンダ

- 『SQL実践入門』の様々なトピックのうち、書いた後に説明が足りないと感じたこと、調査が進んで補足が必要になったこと、仮説にとどまるので本には書けなかったことなどを中心にお話しします。

集合指向言語における手続き型の復権

集合指向言語としてのSQLがテーマ

(2008)



手続き型の復権がテーマ

(2015)

本書の主役はみんなの脇役

- ✓本書の最大の目的はウィンドウ関数を広めること。
- ✓実行計画も解説もしているが、どちらかというと、なぜウィンドウ関数が素晴らしいかを理解するための準備。別に読者に変態的な実行計画を読めるようになってほしいわけではない。
- ✓素晴らしく便利なのに、イマイチ有効利用されない期間が長かった。なぜだろうか。MySQL「なんでやろなあ」

ウィンドウ関数の分かりにくさ

＿人人人人人人人人人人人人人人＿
＞　ウィンドウってなんだよ　＜
￣Y^Y^Y^Y^Y^Y^Y^Y^Y^Y^Y^Y^Y^Y￣

実際、一般的によく見るウィンドウ関数の構文には、ウィンドウの定義が存在しないように見える。

```
SELECT shohin_id, shohin_mei, hanbai_tanka,
     AVG (hanbai_tanka) OVER (ORDER BY shohin_id
                              ROWS BETWEEN 2 PRECEDING
                                   AND CURRENT ROW) AS moving_avg
  FROM Shohin;
```

実はよく見る構文は簡略形

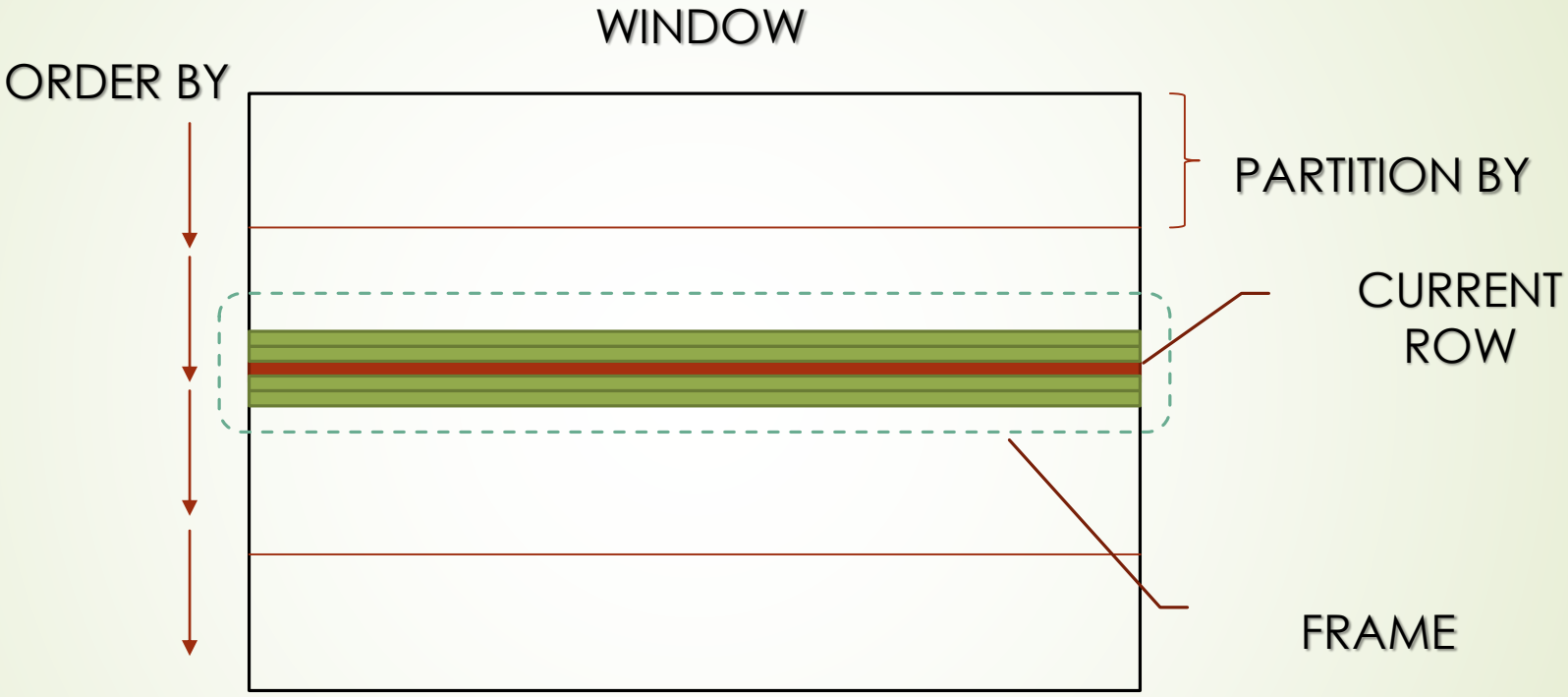
ウィンドウ関数には二つの構文が存在する。普段我々がよく見る構文は、暗黙的に「**無名ウィンドウ**」を使う簡略形。（無名プロシージャとか無名ビューと同じ）

ウィンドウを明示的に定義する構文もある。こちらの**有名ウィンドウ**構文の便利なところは、ウィンドウの使いまわしが可能なこと。（CTEやプロシージャと同じ）

```
SELECT shohin_id, shohin_mei, hanbai_tanka,
       AVG (hanbai_tanka) OVER W AS moving_avg
  FROM Shohin
WINDOW W AS (ORDER BY shohin_id
                 ROWS BETWEEN 2 PRECEDING
                          AND CURRENT ROW);
```

※この正式な構文は、各DBMSのマニュアルには書いてあったりなかったり・・・
文法的に受け付けないDBMSもある。（ex.Oracle）

1ページでわかるウィンドウ関数



以下の論文から少し改変

V.Leis, A. Kemper, K.Kundhikanjana, T.Neumann, 2015, *Proceedings of the VLDB Endowment* Efficient Processing of Window Functions in Analytical SQL Queries

http://www.vldb.org/pvldb/vol8/p1058-leis.pdf

ウィンドウ関数の実行計画

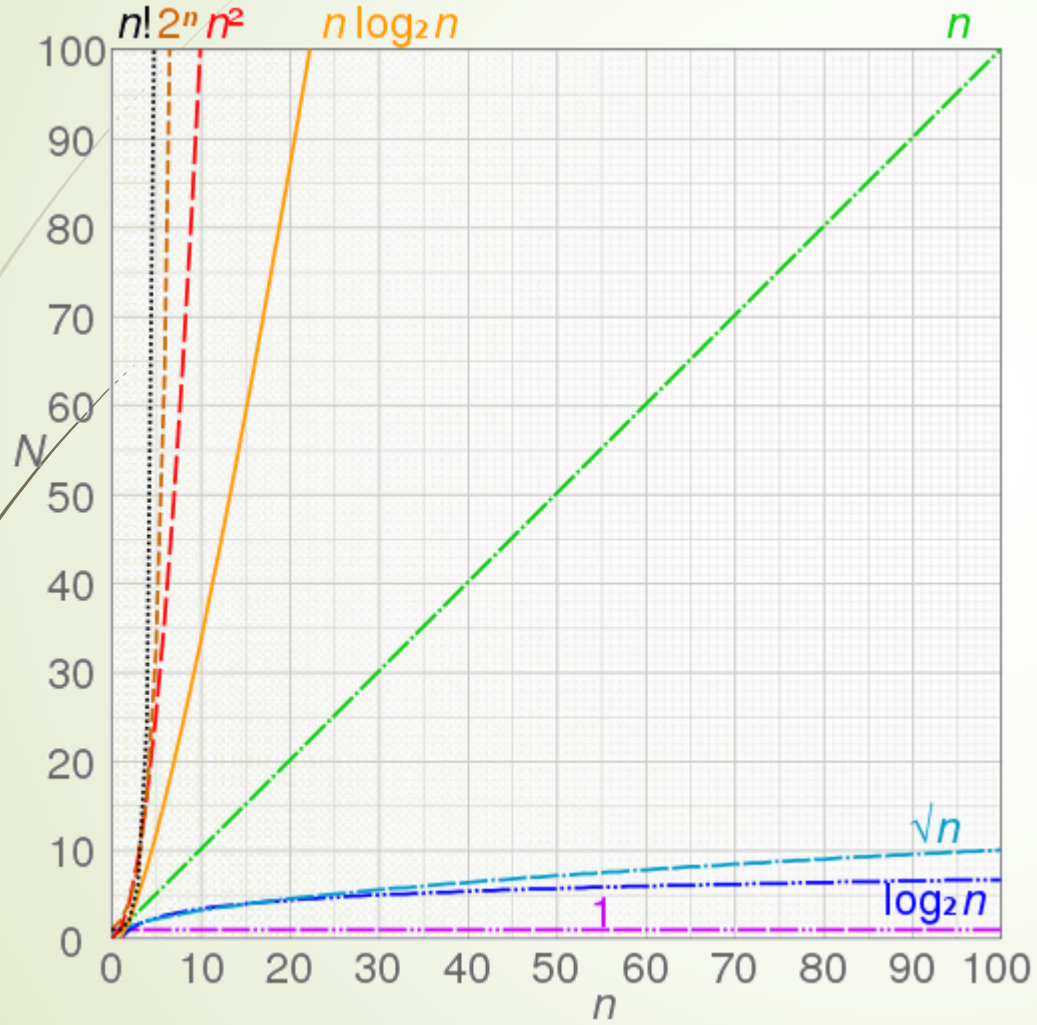
ウィンドウ関数の内部実装は、現在のところ例外なくソートで実現されている。しかし、原理的にはPARTITION BYはGROUP BYと同じくHASHで実装することも可能。前頁の論文も、"理論的には"その方がパフォーマンスが有利としている。入力行数nに対してパーティション数が$O(n)$だとすれば、HASHは$O(n)$、ソートは最善でも$O(n \log n)$になる。

（なお、どのみちOVER句でORDER BYを使うことが多いから、ソートは必要になるので、両者の性能差はそんなに綺麗には出ない、という点も指摘されている）。

ウィンドウ関数の実行計画（PostgreSQL）

```
QUERY PLAN
------------------------------------------------------------------------
 WindowAgg  (cost=20.76..24.61 rows=220 width=242)
   ->  Sort  (cost=20.76..21.31 rows=220 width=242)
         Sort Key: shohin_id
         ->  Seq Scan on shohin  (cost=0.00..12.20 rows=220 width=242)
```

[参考]計算量のグラフ



https://en.wikipedia.org/wiki/Big_O_notation

※余談だが、このグラフは結合アルゴリズムにおいて、SORT MERGEがめったにHASHに勝てない理由も示している。単純化してテーブルの行数が両方nとすると、

HASH：O(*n*)
SORT MERGE：O(*n* log *n*)

【クイズ】
Nested Loopは素朴に考えるとO(n^2)で割と最悪な計算量に見えるが、それでも結合アルゴリズムのファーストチョイスなのはなぜでしょう？ (本書第6章を読もう)

相関サブクエリを消去する

Shohin

shohin_id	shohin_mei	shohin_bunrui	hanbai_tanka
0001	Tシャツ	衣服	1000
0002	穴あけパンチ	事務用品	500
0003	カッターシャツ	衣服	4000
0004	包丁	キッチン用品	3000
0005	圧力鍋	キッチン用品	6800
0006	フォーク	キッチン用品	500
0007	おろしがね	キッチン用品	880
0008	ボールペン	事務用品	100

相関サブクエリ

```
SELECT shohin_bunrui, shohin_mei, hanbai_tanka
  FROM Shohin S1
 WHERE hanbai_tanka >
       (SELECT AVG(hanbai_tanka)
          FROM Shohin S2
         WHERE S1.shohin_bunrui = S2.shohin_bunrui
         GROUP BY shohin_bunrui);
```

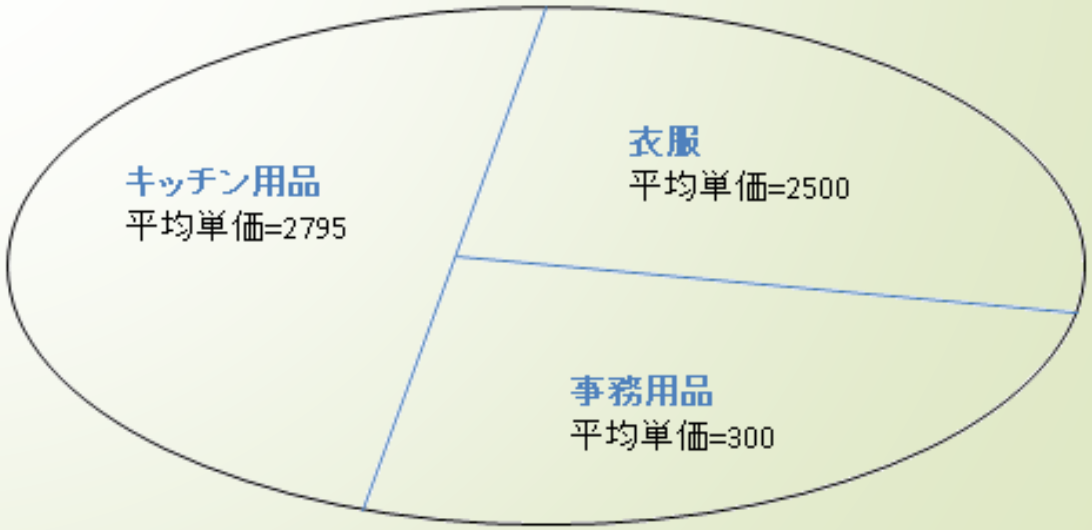
ウィンドウ関数

```
SELECT shohin_bunrui, shohin_mei, hanbai_tanka
  FROM (SELECT shohin_mei, shohin_bunrui, hanbai_tanka,
               AVG(hanbai_tanka)
                 OVER(PARTITION BY shohin_bunrui) AS avg_tanka
          FROM Shohin) TMP
 WHERE hanbai_tanka > avg_tanka;
```

キーワードは集合の”カット”

```
SELECT 衣服,          Tシャツ,          1000 FROM Shohin WHERE 1000 > 2500;
SELECT 衣服,          カッターシャツ,   1000 FROM Shohin WHERE 4000 > 2500;
-------------------------------------------------------------------------------------------------
SELECT キッチン用品, 包丁,              3000 FROM Shohin WHERE 3000 > 2795;
SELECT キッチン用品, 圧力鍋,            6800 FROM Shohin WHERE 6800 > 2795;
SELECT キッチン用品, フォーク,           500 FROM Shohin WHERE  500 > 2795;
SELECT キッチン用品, おろしがね,         880 FROM Shohin WHERE  880 > 2795;
-------------------------------------------------------------------------------------------------
SELECT 事務用品,      ボールペン,       100 FROM Shohin WHERE  100 > 300;
SELECT 事務用品,      穴あけパンチ,     500 FROM Shohin WHERE  500 > 300;
```

相関サブクエリもウィンドウ関数も集合のカットを行うという点で同じ機能を持つ。



相関サブクエリの消去は利点がいっぱい

- ✓ コードの可読性があがる。
- ✓ テーブルへのアクセスが1回ですむ。
- ✓ 実行計画が単純になり性能が安定する。
- ✓ ウィンドウ関数には今後性能改善の期待が持てる。

ウィンドウ関数に相関サブクエリを消去する効果があることには、早くから気付いている人はいた。その劇的な効果は”WinMagic”と呼ばれた。（なお流行らなかった模様）

C.Zuzarte , H.Pirahesh , W.Ma , Q.Cheng , L.Liu , K.Wong,
WinMagic: subquery elimination using window aggregation, *ACM SIGMOD*, 2003
https://www.researchgate.net/publication/221214692_WinMagic_Subquery_Elimination_Using_Window_Aggregation

[余談]GROUP BYを消去することもできる

Shohin

shohin_id	shohin_mei	shohin_bunrui	hanbai_tanka
0001	Tシャツ	衣服	1000
0002	穴あけパンチ	事務用品	500
0003	カッターシャツ	衣服	4000
0004	包丁	キッチン用品	3000
0005	圧力鍋	キッチン用品	6800
0006	フォーク	キッチン用品	500
0007	おろしがね	キッチン用品	880
0008	ボールペン	事務用品	100

GROUP BY

```
SELECT shohin_bunrui, COUNT(*)
  FROM Shohin
 GROUP BY shohin_bunrui
HAVING COUNT(*) > 2;
```

ウィンドウ関数

```
SELECT shohin_bunrui, shohin_mei, hanbai_tanka, cnt
  FROM (SELECT shohin_mei, shohin_bunrui, hanbai_tanka,
               COUNT(*)
                 OVER(PARTITION BY shohin_bunrui) AS cnt
          FROM Shohin) TMP
 WHERE cnt > 2;
```

ウィンドウ関数小史

- 1999　Oracle 8iがウィンドウ関数をサポート
  OracleとIBMが共同でANSIに標準化を提案
  SQL:1999に（オプションの）修正として採択される
- 2000　IBMがDB2 LUWでサポート
- 2003　SQL:2003にフルレベルで標準化される
- 2005　マイクロソフトがSQL Server 2005 でサポート
- 2009　PostgreSQL が8.4でサポート
- 2011　SQL:2011でframe構文が標準化される
- 2015　本書刊行
- 2017　MySQL が8.0.2でサポート

(HAPPY) END

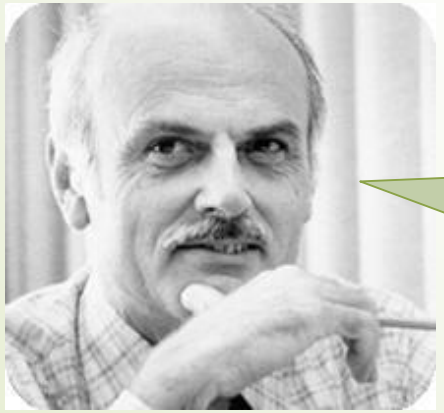
・・・ふとした疑問が

なぜウィンドウ関数の登場は
こんなにも遅かったのだろう。
すごく便利だし、
内部実装もレコードをソート
するだけだから、やろうと思えば
SQLの初期から実装できたのでは？

仮説：RDBのセントラルドグマ

“The n-ary relation was chosen as the single aggregate structure for the relational model, because with appropriate operators and an appropriate conceptual representation (the table) it satisfies all three of the cited objectives. Note that an n-ary retation is a mathematical set, in which **the ordering of rows is immaterial**.**”**

The 1981 ACM Turing Award Lecture
Relational Database: A Practical Foundation for Productivity
http://amturing.acm.org/award_winners/codd_1000892.cfm



行の順序は（重要では）ない

リレーショナル原理主義右派の言い分

“*No row ordering*: There shouldn’t be any disagreement over this one, even though it isn’t usually mentioned in the context of normalization as such.**”**

Chris Date, Date on Database: Writings 2000-2006, 2007
https://books.google.co.jp/books?id=y_eVBB5qdwMC

行に順序などない
異論は認めない

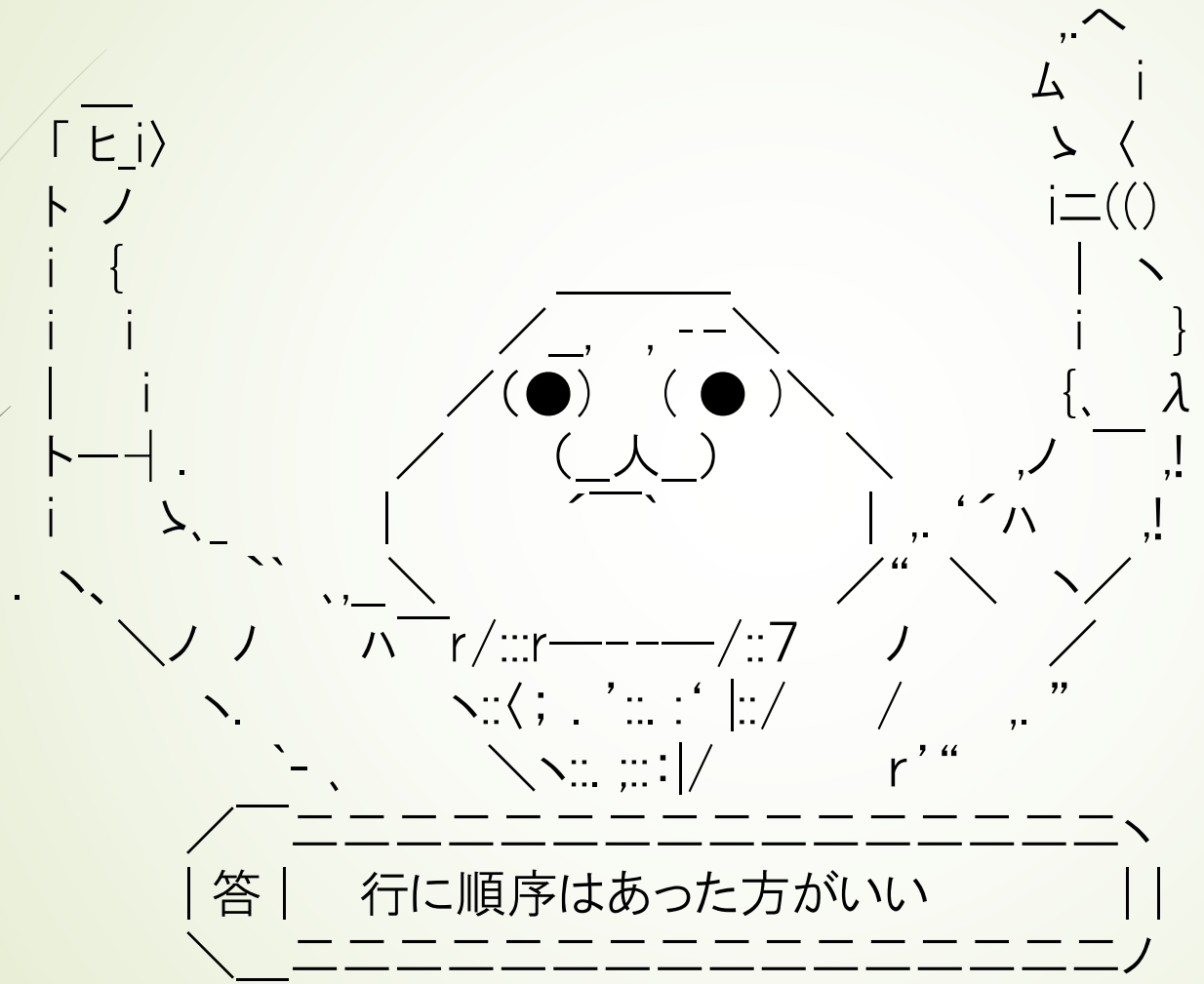
リレーショナル無政府主義左派の言い分

“One of the deficiencies of SQL is the lack of support for analytic calculations like moving averages, cumulative sums, ranking, percentile and lead and lag which are critical for OLAP applications. These functions work on **ordered sets of data**. Expressing these functions in current SQL is not elegant, requires self-joins and is difficult to optimize..”

S.Bellamkonda, T.Bozkaya, B.Ghosh, A.Gupta, J.Haydu, S.Subramanian, A.Witkowski, 1999, *Analytic Functions in Oracle 8i*
http://infolab.stanford.edu/infoseminar/archive/SpringY2000/speakers/agupta/paper.pdf

行に順序はありまあす♪

正解



手続き型には二種類ある。良い手続き型と悪い手続き型だ。

ウィンドウ関数によって、手続き型の良い点を取りこめたが、根強く残る手続き型の旧弊も。その代表が

ぐるぐる系

お前は復活せんでいい

強力すぎる心理モデル

- ✓ ぐるぐる系がなくならないのは、ファイルとループという手続き型の心理モデルがあまりに強力すぎるから。
- ✓ 性能さえ問題ないなら、多くの利点があることもまた事実なのだが・・・。
- ✓ こいつのチューニングは石をヤスリ掛けするがごとき苦行か抜本的な処理方式の見直しになるので、できればあまり経験したくない。

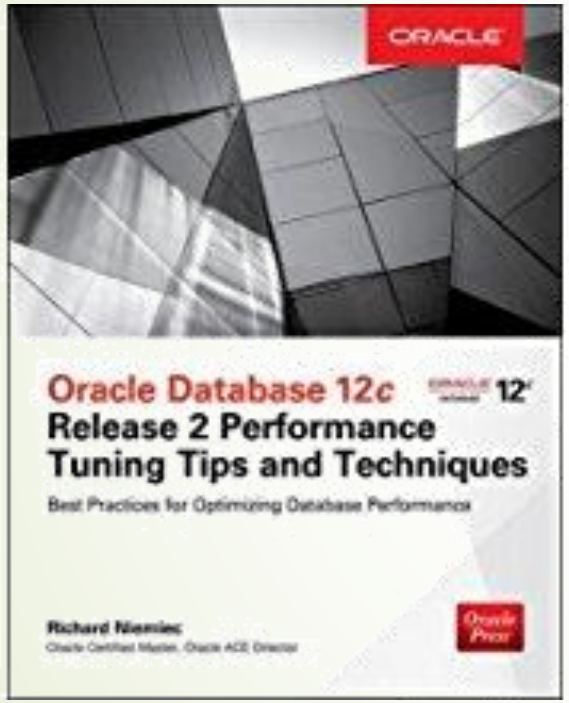
若山 勝吾：SQL大量発行処理をいかにして高速化するか

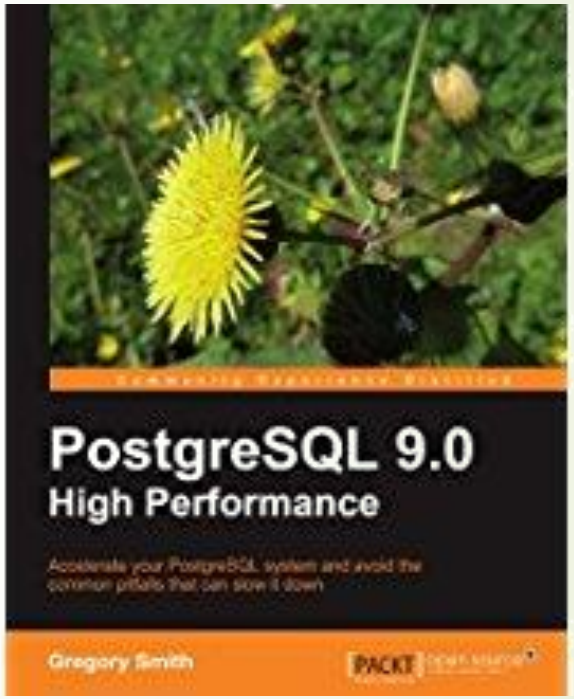
https://www.slideshare.net/ShogoWakayama/sql-79530929

ぐるぐる系オーバーヘッド一覧

- NW送受信によるラウンドトリップ遅延
- DBに対するコネクションの確立／切断
- SQL文のパース
- ストレージI/O
- SQL文における文字列・型変換、計算などの関数
- アプリケーション側のビジネスロジック処理
- REDOログ書き込み競合

もっとRDBのパフォーマンスを追求したいあなたのための参考文献





※本書の種本です

最後のお願い

DBチューニングが
不要になる
世界を目指して

(TRUE)
END