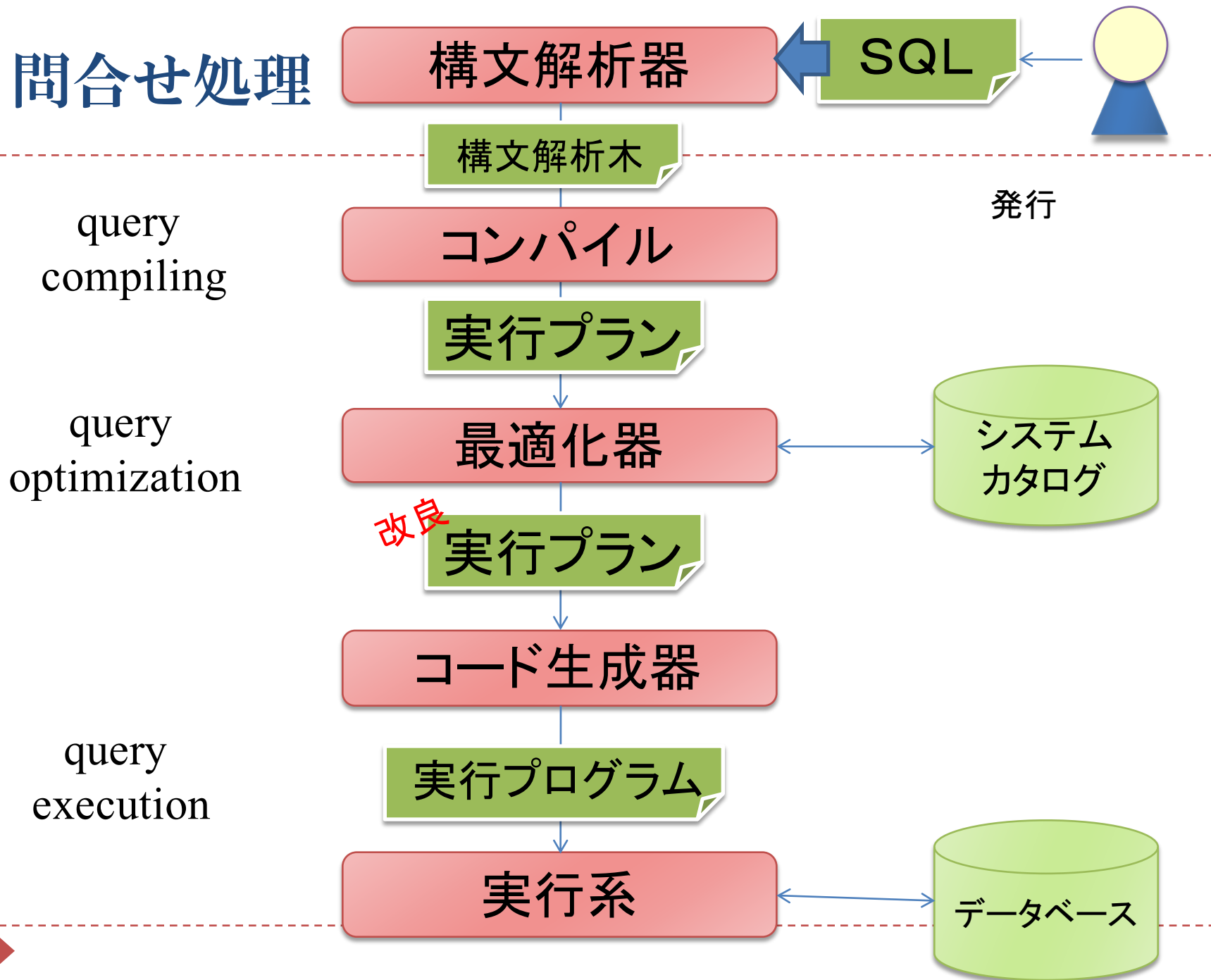


データベースシステム

第9回 問合せプラン構成

2016年7月5日

問合せ処理



問合せ処理の流れ（例に沿って）

Sailors (sid, sname, rating, age)

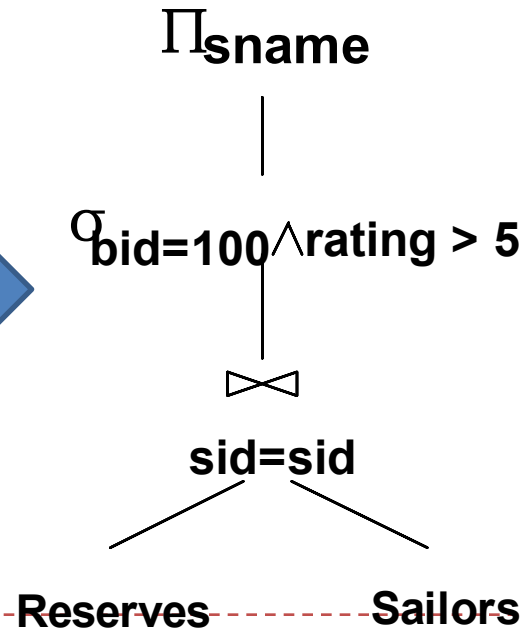
Reserves (sid, bid, day, rname)

```
SELECT S.sname
FROM Reserves R, Sailors S
WHERE R.sid=S.sid AND R.bid=100
AND S.rating>5
```

問合せ実行プラン

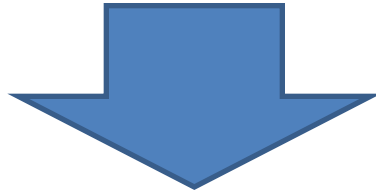
$\pi_{\text{sname}}(\sigma_{\text{bid}=100 \wedge \text{rating} > 5}(\text{Reserves} \bowtie \text{Sailors}))$

関係代数の形に直す



問合せ実行プラン

- ▶ 基本処理(選択、射影、結合)の組合せ
- ▶ 下から順番に処理する
- ▶ はじめは素直に関係代数の内側から処理する順番で問合せ実行プランを作る



問合せ最適化(Query Optimization)

もっと速い問合せ処理を実行するための実行プランに書き換える

最適化器 = Optimizer

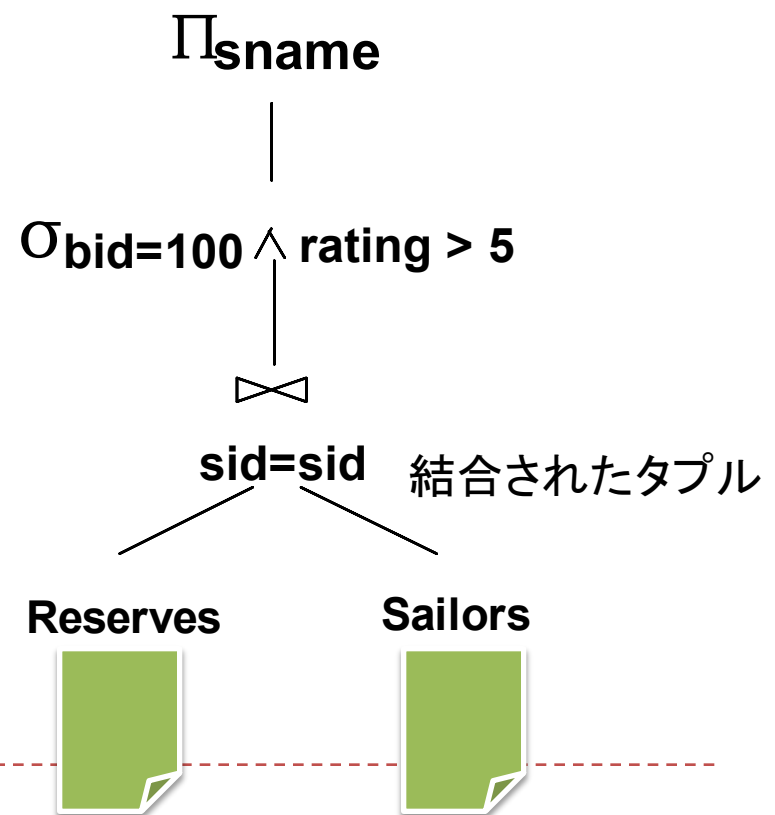
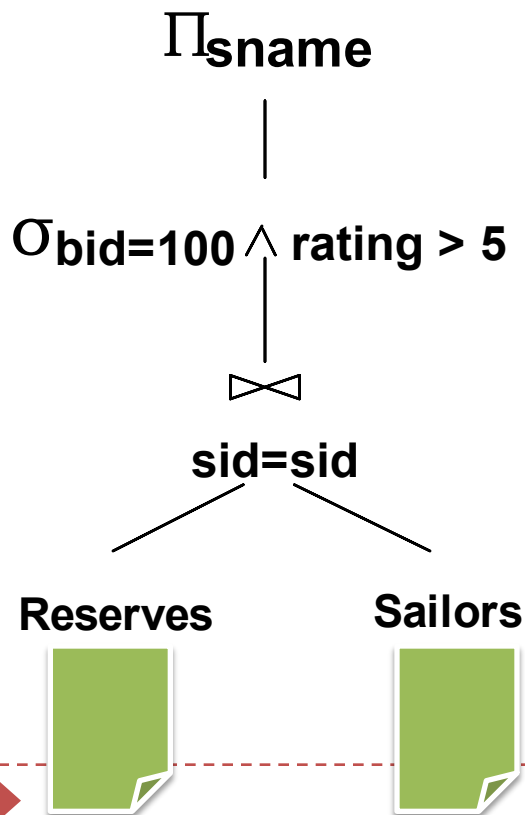


問合せの実行

on-the-fly

(1) 処理毎に中間テーブルを作る

(2) 流れ作業のように前の演算で処理の終わったタプルを直ぐに受取り実行していく



問合せ最適化

▶ 実行プランの各々の処理を どんな方法で実行するか

▶ 選択

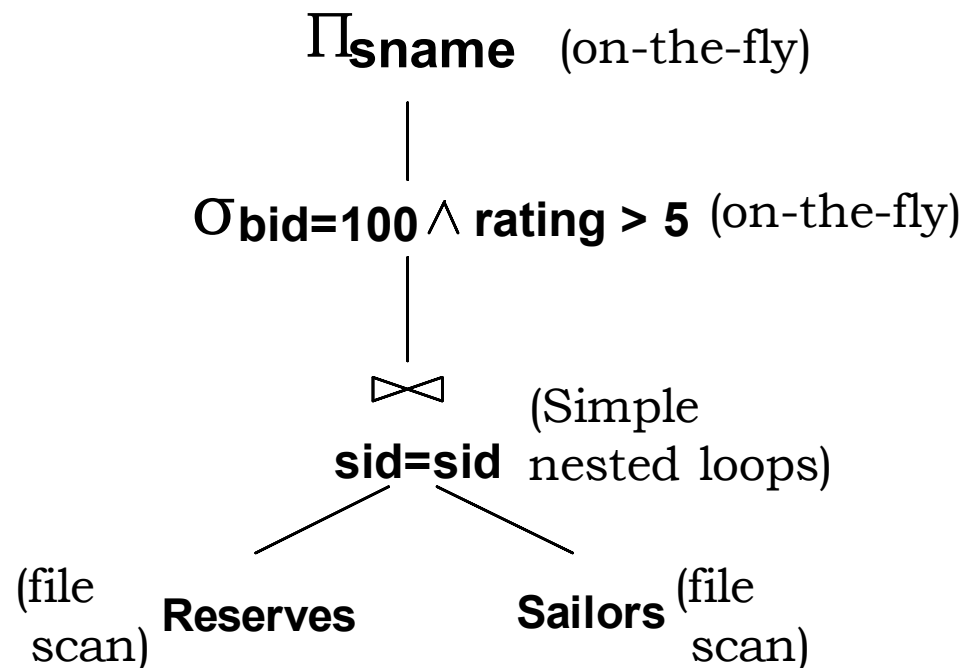
- ▶ スキャンする
- ▶ 索引を使う

▶ 射影

▶ 結合

- ▶ nested loop
- ▶ sort-merge join
- ▶ hash join

▶ どの順番で実行するか



問合せ最適化の必要性（直観的な説明）

通常の実行計画の場合

Table A

ID	value
1	1
2	2
...	...
10000000	

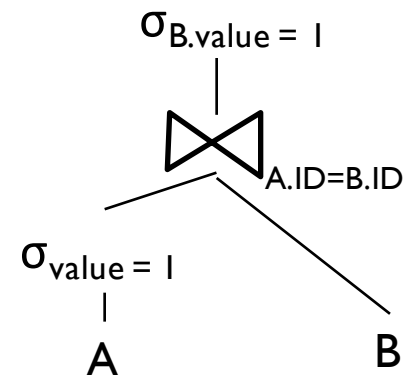
Table B

ID	value
1	1
2	2
...	...
100	100

発行されたSQL

```
SELECT *  
FROM A,B  
WHERE A.ID=B.ID  
      and A.value = 1  
      and B.value = 1
```

考えられる実行プラン



問合せ最適化の必要性（直観的な説明）

前ページの実行プランがうまくいかないケース

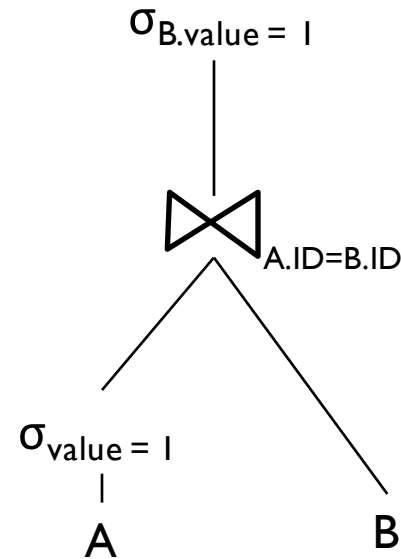
→ Table Aの8割がvalue=1である場合

Table A

ID	value
1	1
2	1
...	...
10000000	1

Table B

ID	value
1	1
2	2
...	...
100	100



問合せ最適化の必要性（直観的な説明）

前ページの場合の適切な実行プランを使うと...

Table B

ID	value
1	1
2	2
...	...
100	100

Table A

ID	value
1	1
2	1
...	...
10000000	1

問合せ最適化

▶ 最適な実行プランを作るために考えること

1. 実行する演算の順序(木の組み換え)
2. それぞれの演算で何のアルゴリズムを採用するか(索引を使う・使わない、など)

▶ どうやって最適なものを見つけるのか

1. 他のプランをいくつか挙げていく
2. それぞれのプランのコスト見積もりをする
3. コストが小さいものを選択する



演算のコストを見積もるために

- ▶ 演算のコストを見積もるためにはいろいろな情報が必要となる
 - ▶ テーブルのデータ量、最大値、最小値、値の分布
 - ▶ 索引がどの属性に貼られているか
- ▶ それらの情報はどこに行けばアクセスできるのか？



システムカタログ

リレーショナルデータベースの管理システムがテーブルや列の情報などのスキーマメタデータと内部的な情報を格納する場所

システムカタログ

- ▶ テーブル情報
 - ▶ テーブル名、ファイル名、ファイル構造
 - ▶ 属性名とデータ型
 - ▶ 索引名
 - ▶ 整合性制約(主キーや外部キー)
 - ▶ 各索引に関して
 - ▶ 索引名と構造(B+-treeなど)
 - ▶ 検索キー属性
 - ▶ 各ビューに関して
 - ▶ ビュー名と定義
 - ▶ 統計情報
-



代表的な統計情報

- ▶ Cardinality :
テーブルRにおけるタプル数
- ▶ Size:
テーブルRにおけるページ数
- ▶ Index Cardinality:
索引Iにおけるキー値の数
- ▶ Index Size:
索引のページ数
- ▶ Index Height:
木索引の高さ
- ▶ Index Range:
索引中のキー値の最小値と最大値



カテゴリの格納の仕方

▶ システム用のテーブルとして管理する

attr_name	rel_name	type	position
sid	Sailors	integer	1
sname	Sailors	string	2
rating	Sailors	integer	3
age	Sailors	integer	4
sid	Reservers	integer	1
bid	Reservers	integer	2
day	Reservers	dates	3
rname	Reservers	string	4



問合せ最適化の例

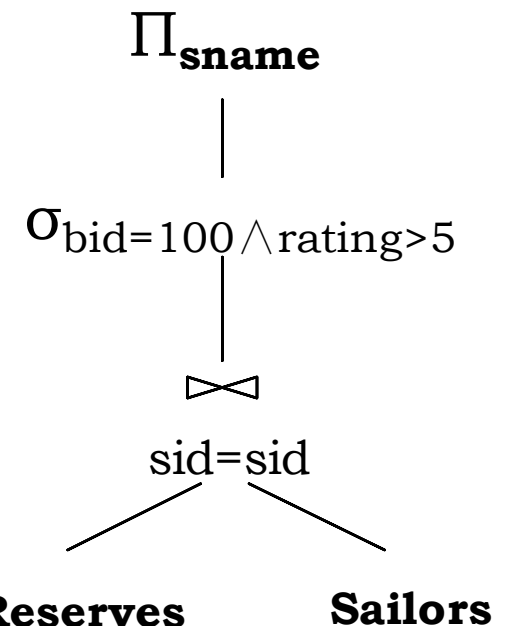
Sailors (sid, sname, rating, age)

Reserves (sid, bid, day, rname)

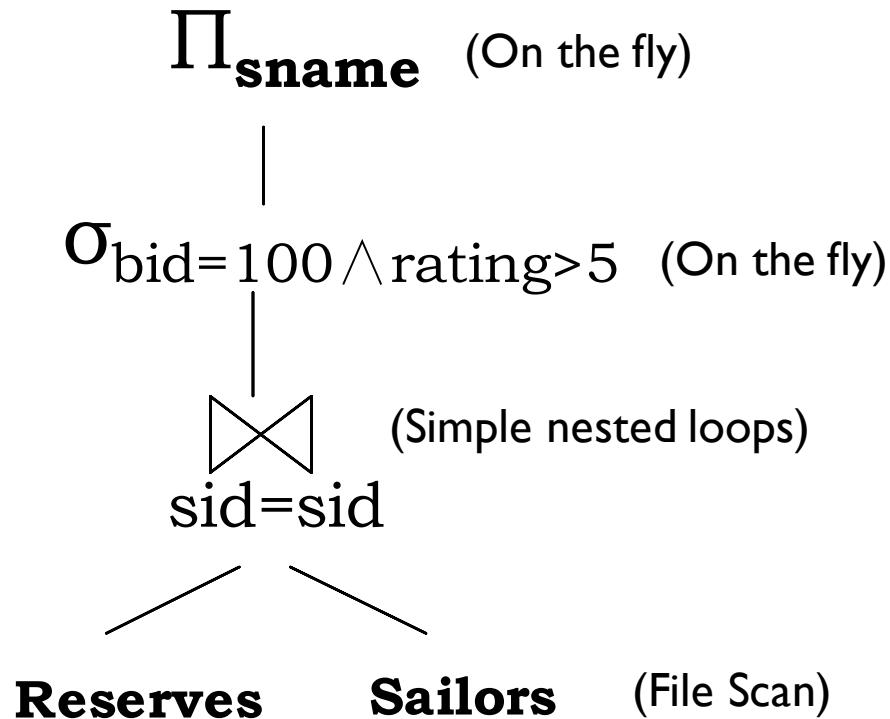
```
SELECT S.sname
FROM Reserves R, Sailors S
WHERE R.sid=S.sid AND R.bid=100
AND S.rating>5
```

Table	タプル長 (bytes)	1ページ内 のレコード数	レコード 数	ページ数
Sailors	40	100	100000	1000
Reservers	50	80	40000	500

問合せ実行プラン

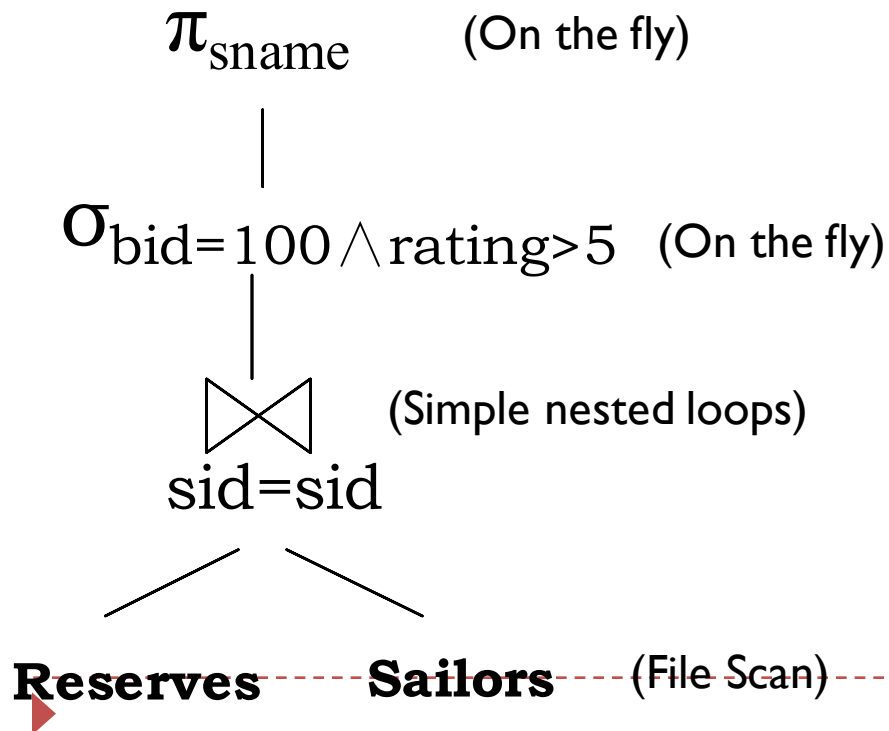


まずは元のプラン木のコストを見積もってみる



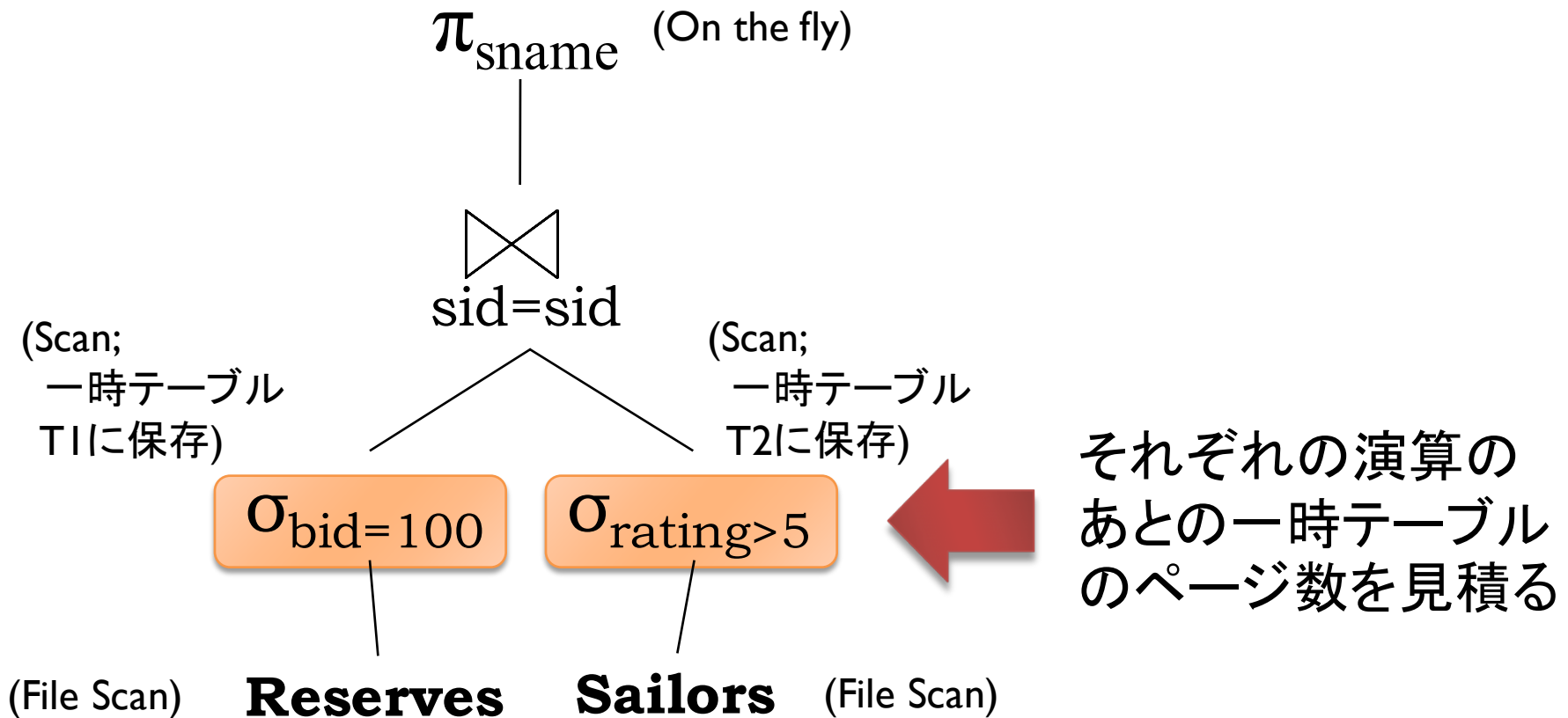
コスト計算をしてみよう

Table	タプル長 (bytes)	1ページ内 のレコード数	レコード 数	ページ数
Sailors	40	100	100000	1000
Reservers	50	80	40000	500



代替実行プラン案（その1）

▶ Selectionを上にあげる



選択されるタプル数の見積もり

まず選択率を見積もる

- ▶ $column = value$ の場合
 - ▶ $column$ が索引付き... $1/(\text{キーの数})$
 - ▶ $column$ が索引なし... $1/10$
- ▶ $column > value$ の場合
 - ▶ $column$ が索引付き...
 $(\text{キーの最大値} - value) / (\text{最大値} - \text{最小値})$
 - ▶ $column$ が索引なし... 半分弱
- ▶ $column1 > column2$
 - ▶ 両 $column$ とともに索引付き
 $1/\text{Max}(\{\text{column1 のキー数}, \text{column2 のキー数}\})$



選択率から一時テーブルの大きさを求める

$\sigma_{\text{bid}=100}$

(File Scan)

Reserves

Reserver.bidは索引| bidindexが
付いているとする

bidindexのキー数: 100

$\sigma_{\text{rating}>5}$

Sailors (File Scan)

Sailors.rating は索引| rtindexが
付いているとする

rtindexのキーの最大値: 10

rtindexのキーの最小値: 1

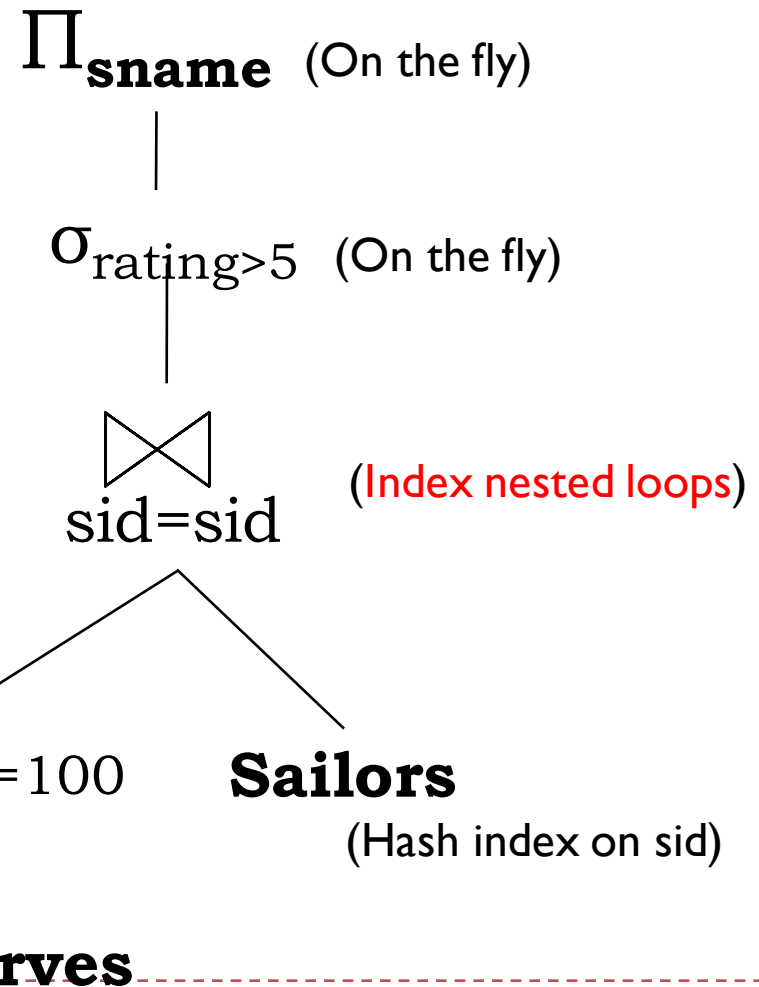


代替実行プラン案（その2）

▶ 索引を使う

Reserver.bidの索引 bidindexは
ハッシュ索引(一次索引)と
する

Sailors.sidの索引 sidindexも
ハッシュ索引(一次索引)



(Hash index on bid)

Reserves

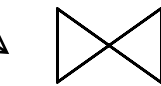
代替実行プラン案（その2）

▶ 索引を使う

左下から送られたタプルのsidからハッシュ値を求め、対応するSailorsのタプルを検索して結合

$\Pi_{\mathbf{sname}}$ (On the fly)

$\sigma_{\text{rating}>5}$ (On the fly)



sid=sid

(Index nested loops)

bidのハッシュ索引を使う

$\sigma_{\text{bid}=100}$

Sailors

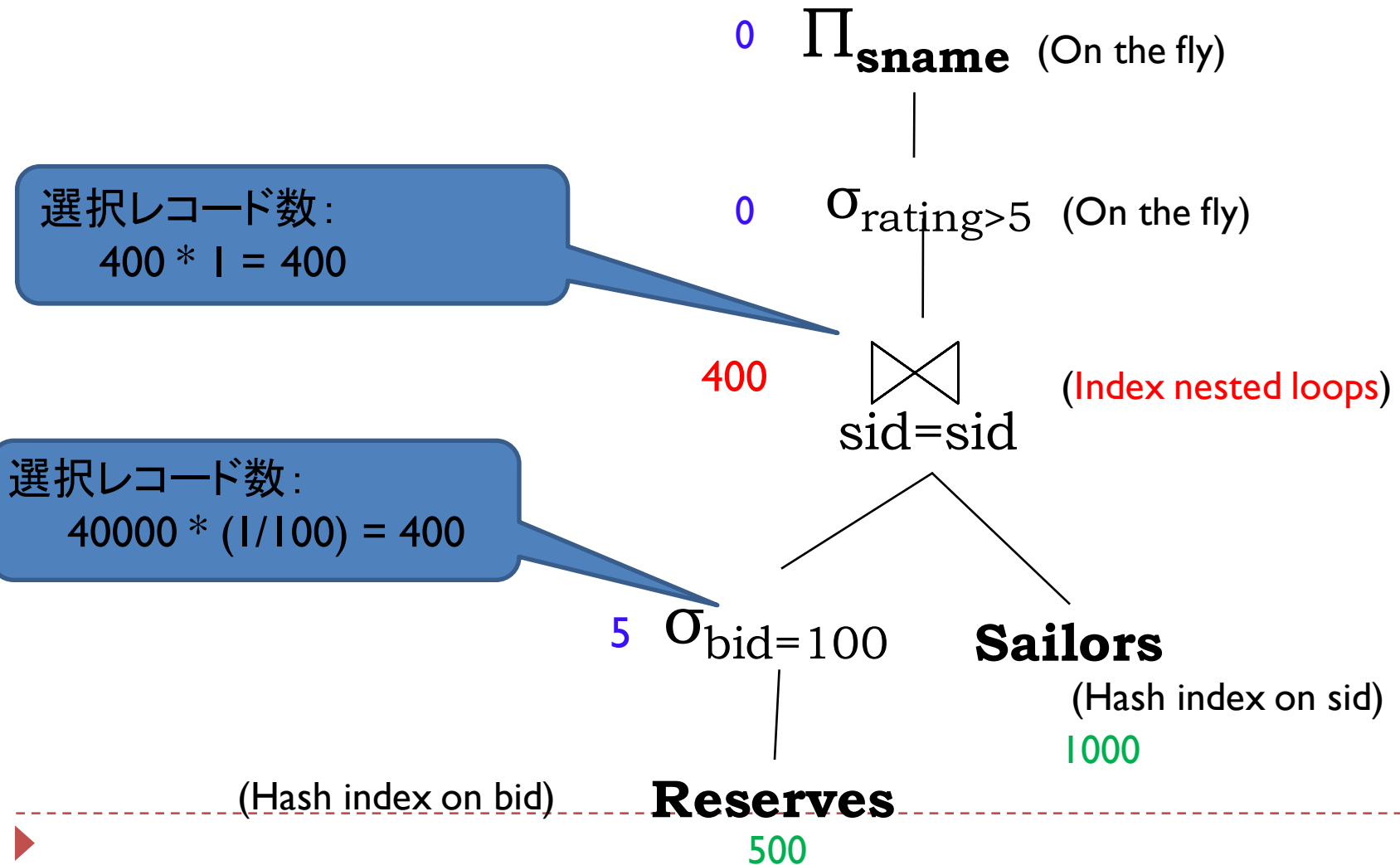
(Hash index on sid)

(Hash index on bid)

Reserves

代替実行プラン案 (その2)

▶ コストを計算してみよう



関係代数演算の等値関係

- ▶ 代替プラン木を組み立てる時に利用する

$$\sigma_{c1 \wedge c2 \wedge \dots \wedge cn}(R) = \sigma_{c1}(\sigma_{c2}(\dots(\sigma_{cn}(R))\dots))$$

$$\sigma_{c1}(\sigma_{c2}(R)) = \sigma_{c2}(\sigma_{c1}(R))$$

$$\pi_{a1}(R) = \pi_{a1}(\pi_{a2}(\dots(\pi_{an}(R))\dots))$$

$$\text{但し, } a_i \subseteq a_{i+1}$$

$$R \times S = S \times R$$

$$R \bowtie S = S \bowtie R$$



関係代数演算の等値関係

$$\pi_a(\sigma_c(R)) \equiv \sigma_c(\pi_a(R))$$

※ただし属性リストaの中に探索条件cで使う属性が含まれているとする

$$R \bowtie_c S \equiv \sigma_c(R \times S)$$

$$\pi_a(R \times S) \equiv \pi_{a_1}(R) \times \pi_{a_1}(S)$$

$$\pi_a(R \times_c S) \equiv \pi_{a_1}(R) \times_c \pi_{a_1}(S)$$

$$\pi_a(R \times_c S) \equiv \pi_a(\pi_{a_1}(R) \times_c \pi_{a_1}(S))$$

