Otimização de Consultas Relacionais

Capítulo 15

Otimização de Consultas

- É o processo de selecionar o plano de avaliação de consulta mais eficiente para uma consulta.
- Tipicamente, existem muitos métodos de execução de consultas que provêm as mesmas respostas.
- Custo dos métodos alternativos variam enormemente.
- Desejável que se encontre a estratégia de menor custo de execução.
  - Iremos abordar:
  - Equivalência da álgebra relacional.
  - Estimativas de custo.
    - Tamanho do resultado estimado e fator de redução.
    - Estatísticas e catálogos.
  - Enumeração de planos alternativos.

Destaques do Otimizador do Sistema R

- Impacto:
  - Mais amplamente utilizado hoje em dia; trabalha bem para um número de junções inferior a 10.
- Estimativa de Custo: Melhor aproximação.
  - Estatísticas, mantidas em catálogos do sistema, são usadas para estimar custo de operações e tamanhos de resultados.
  - Considera uma combinação de custos de CPU e E/S.
- Espaço de Planos: Muito grande, precisa ser podado.
  - Somente o espaço de *planos de profundidade à esquerda* é considerado.
    - Planos de profundidade à esquerda permitem que a saída de cada operador seja encadeada ao próximo operador sem armazená-la em uma relação temporária.
  - Evitam-se produtos cartesianos.

Visão Geral de Otimização de Consultas

- *Plano: Árvores de ops da A.R., com escolha de alg para cada op.*
  - Cada operador é tipicamente implementado usando uma interface do tipo `**empurra'**: quando um operador é `**empurrado'** para as próximas tuplas de saída, ele `**empurra'** suas entradas e as calcula.
- Dois pontos principais:
  - Para uma certa consulta, **quais planos são considerados**?
    - Algoritmo para procurar o plano mais barato (estimado) no espaço de planos.
  - Como é **estimado o custo de um plano**?
- Idealmente: Quer-se encontrar o melhor plano. Na prática: Evitar os piores planos!
- Estudaremos a abordagem do Sistema R.

Esquema para os Exemplos

Sailors (*sid*: integer, *sname*: string, *rating*: integer, *age*: real)
Reserves (*sid*: integer, *bid*: integer, *day*: dates, *rname*: string)

- Similar ao esquema anterior; *rname* adicionada para variações.
- Reserves:
  - Cada tupla tem 40 bytes, 100 tuplas por página, 1000 páginas.
- Sailors:
  - Cada tupla tem 50 bytes, 80 tuplas por página, 500 páginas.

Blocos de Consulta: Unidades de Otimização

- Uma consulta SQL é compilada em uma coleção de ***blocos de consulta***, e estas são otimizadas um bloco por vez.
- Blocos aninhados são geralmente tratados como uma chamada a uma sub-rotina, feita uma vez por tupla mais externa. (É uma simplificação exagerada, mas serve por enquanto.)

```
SELECT  S.sname
FROM  Sailors S
WHERE  S.age IN
     (SELECT  MAX (S2.age)
      FROM  Sailors S2
      GROUP BY  S2.rating)
```

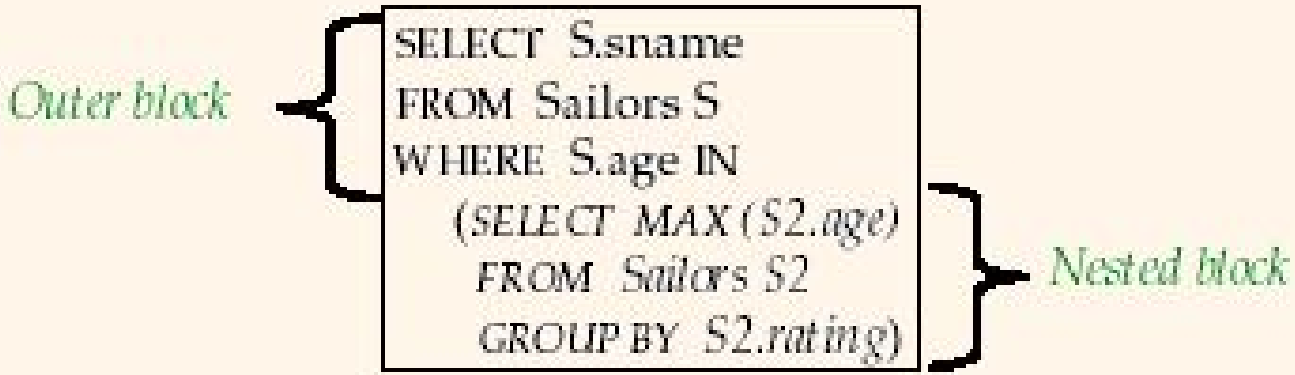
Bloco mais externo

Bloco aninhado

- Cada bloco será convertido pela álgebra relacional.
- Para cada bloco, os planos considerados são:
  - Todos os métodos de acesso disponíveis, para cada relação na cláusula FROM.
  - Todas as ***árvores de junção de profundidade à esquerda***
  - Plano com o menor custo estimado será selecionado.

Passo1: Dividir a consulta em blocos

- **Query Block** = Unidade de otimização
- **Nested Block**: usualmente tratados como chamada de subrotina feita uma vez por tupla extena.



Passo2: Converter o bloco de consulta para a Álgebra Relacional

```
SELECT S.sid
FROM Sailors S, Reserves R, Boats B
WHERE S.sid = R.sid AND R.bid = B.bid AND B.color = "red"
```

$$\pi_{S.sid}(\sigma_{B.color = \text{"red"}}(\text{Sailors} \bowtie \text{Reserves} \bowtie \text{Boats}))$$

Exemplo traduzido para a Álgebra Relacional

```
SELECT S.sid, MIN (R.day)
FROM Sailors S, Reserves R, Boats B
WHERE S.sid = R.sid AND R.bid = B.bid AND B.color = "red"
AND S.rating = ( SELECT MAX (S2.rating) FROM Sailors S2)
GROUP BY S.sid
HAVING COUNT (*) >= 2
```

Inner Block

$$\pi_{\text{S.sid, MIN(R.day)}} (\text{HAVING}_{\text{COUNT(*)}>2} (\text{GROUP BY}_{\text{S.Sid}} (\sigma_{\text{B.color = "red"} \wedge \text{S.rating = val}} (\text{Sailors} \bowtie \text{Reserves} \bowtie \text{Boats}))))$$

Equivalências da Álgebra Relacional

- Permite-nos escolher diferentes ordens de junção e `empurrar' seleções and projeções à frente das junções.
- *Seleções*: $\sigma_{c1 \wedge \ldots \wedge cn}(R) \equiv \sigma_{c1}(\ldots \sigma_{cn}(R))$ *(Cascata)*

  $\sigma_{c1}(\sigma_{c2}(R)) \equiv \sigma_{c2}(\sigma_{c1}(R))$ *(Comutativa)*
- *Projeções*: $\pi_{a1}(R) \equiv \pi_{a1}(\ldots(\pi_{an}(R)))$ *(Cascata)*
- *Junções*: $R \bowtie (S \bowtie T) \equiv (R \bowtie S) \bowtie T$ *(Associativa)*

  $(R \bowtie S) \equiv (S \bowtie R)$ *(Comutativa)*

☞Mostre que: $R \bowtie (S \bowtie T) \equiv (T \bowtie R) \bowtie S$

Exemplos

$$\sigma_{age<18 \wedge rating>5}(Sailors)$$

$$\equiv \sigma_{age<18}(\sigma_{rating>5}(Sailors))$$

$$\equiv \sigma_{rating>5}(\sigma_{age<18}(Sailors))$$

$$\pi_{age,rating}(Sailors) \equiv \pi_{age}(\pi_{rating}(Sailors)) \quad (??)$$

$$\pi_{age,rating}(Sailors) \equiv \pi_{age,rating}(\pi_{age,rating,sid}(Sailors))$$

Outras Equivalências:

- Uma projeção comuta com uma seleção que usa somente os atributos mantidos pela projeção.

$$\pi_{age,\ rating,\ sid}\left(\sigma_{age<18\ \wedge\ rating>5}\left(Sailors\right)\right)$$

$$\leftrightarrow\ \sigma_{age<18\ \wedge\ rating>5}\left(\pi_{age,\ rating,\ sid}\left(Sailors\right)\right)$$

Mais Equivalências

- Seleção entre atributos de dois argumentos de um produto cartesiano converte o produto cartesiano em uma junção.

$$\sigma_{S.sid = R.sid}(\text{Sailors} \times \text{Reserves})$$

$$\Leftrightarrow \text{Sailors} \bowtie_{S.sid = R.sid} \text{Reserves}$$

Mais Equivalências (cont)

- Uma seleção apenas nos atributos de R comuta com R ⋈ S. (i.e., $\sigma(R \bowtie S) \equiv \sigma(R) \bowtie S$)

$$\sigma_{S.age<18}(Sailors \bowtie_{S.sid=R.sid} Reserves)$$

$$\leftrightarrow (\sigma_{S.age<18}(Sailors)) \bowtie_{S.sid=R.sid} Reserves$$

- Da mesma forma, se uma projeção segue uma junção R ⋈ S, podemos `empurrá-la' mantendo os atributos de R (e S) que são necessários à junção mas que são mantidos pela projeção.

$$\pi_{S.sname}(Sailors \bowtie_{S.sid=R.sid} Reserves)$$

$$\leftrightarrow \pi_{S.sname}(\pi_{sname,sid}(Sailors) \bowtie_{S.sid=R.sid} \pi_{sid}(Reserves))$$

Enumeração de Planos Alternativos

- Há dois casos principais:
  - **Planos de uma única relação**
  - **Planos de relações múltiplas**
- Para consultas em uma única relação, as consultas consistem de uma combinação de operadores de seleção, projeção e agregações:
  - Cada caminho de acesso disponível (varredura do arquivo/ índice) é considerado, e aquele com o menor custo estimado é escolhido.
  - As diferentes operações são essencialmente executadas juntamente (e.g., se um índice é usado para uma seleção, a projeção é feita para cada tupla lida, e as tuplas resultantes são *encadeadas* para o cálculo dos agregados).

Estimativa de Custo

- Para cada plano considerado, deve-se estimar o custo:
  - Deve-se estimar o *custo* de cada operação na árvore do plano.
    - Depende das cardinalidades das entradas.
    - Já discutimos como estimar o custo de operações (varredura seqüencial, varredura com índice, junções etc.)
  - Também se deve estimar o *tamanho* do *resultado* para cada operação na árvore!
    - Usar informações sobre as relações de entrada.
    - Para seleções e junções, assumir independência de predicados.

Estatísticas e Catálogos

- Informações necessárias sobre relações e índices.
- Catálogos , tipicamente, contém:
  - # tuplas (NTuplas) e # páginas (Npaginas) por Relação.
  - # nro de chaves distintas (NChaves) por índice.
  - Maior/Menor valor da chave (Máx/Min) por índice.
  - Altura da Chave para cada árvore indexada.
- Estatísticas e catálogos devem ser atualizados periodicamente !

  - É muito caro atualizar todas as vezes que os dados são modificados; Catálogo mantém informações aproximada.
- Informações mais detalhadas (ex, histogramas de valores de campos) são as vezes mantidas.

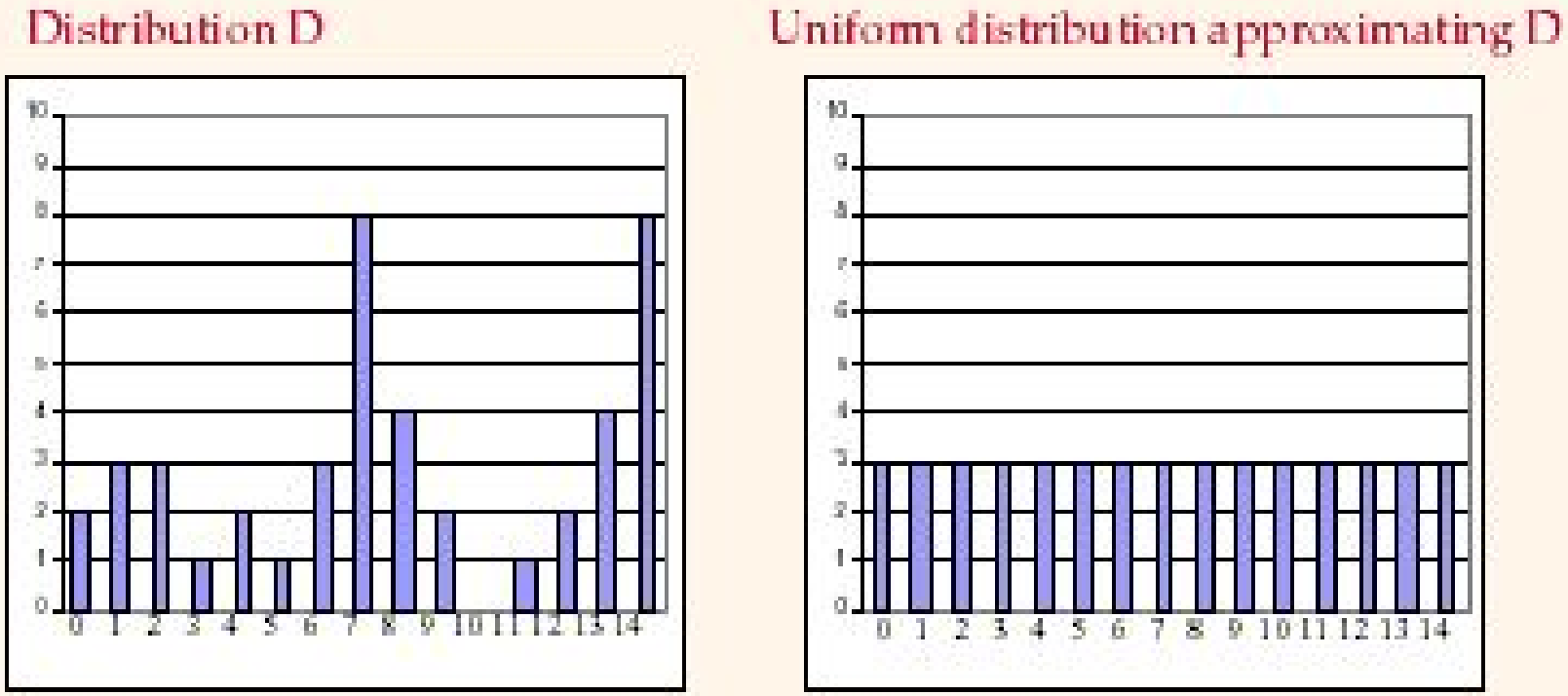
Estimativas de tamanho do resultado da Consulta

- Cardinalidade = (NTuplas) * FR's
- Termo col = valor (dado um índice I na col)
  FR = 1/Nchaves(I)
- Termo col > valor (dado um índice I na col)
  FR = Max(I) – ((valor/Max(I))– Min(I))
- Termo col1 = col2
  FR = 1/Max(NChaves(I_1) ,NChaves(I_2))

- **Ausência de Índice : assumir FR = 1/10**

Histogramas

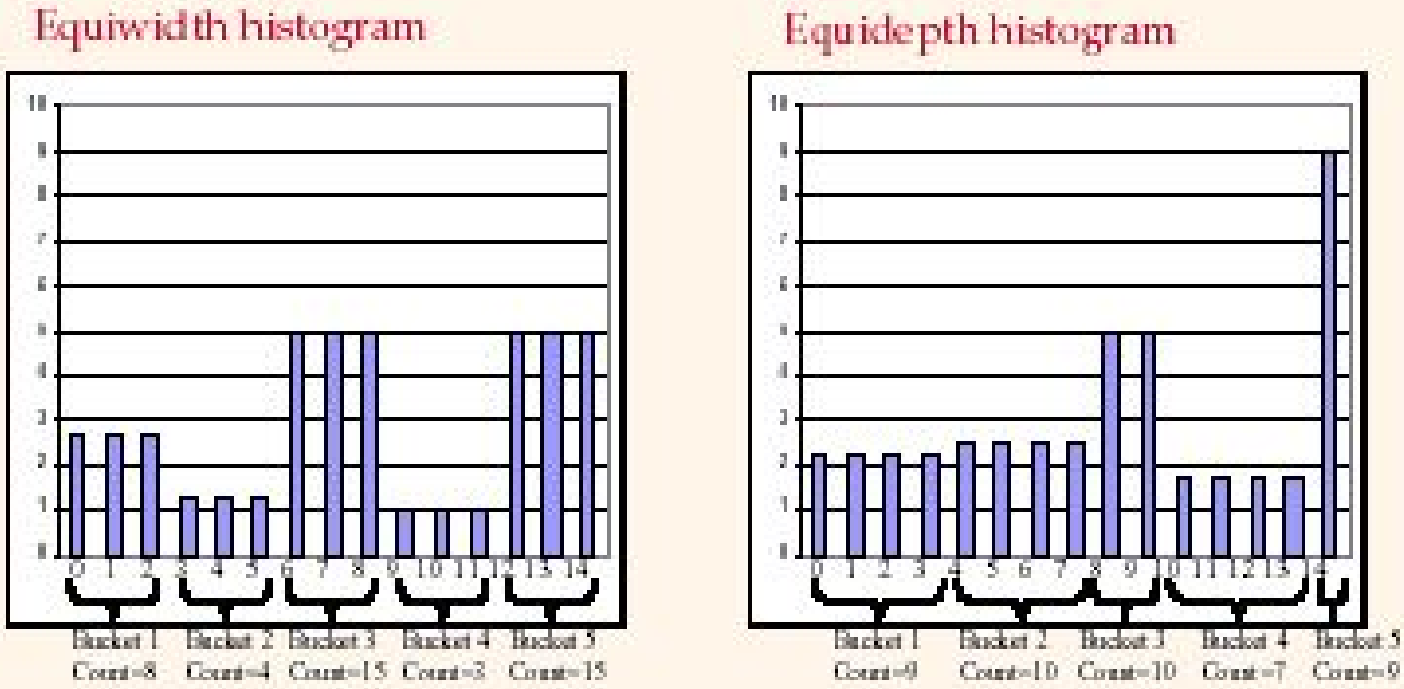
- **Assumir distribuição uniforme é melhor que nada !!**



- **Seleção : age > 13 ???**

Tipos de Histogramas

- Para melhores estimativas, usar histogramas



Planos de uma consulta simples

- **Planos sem índices**:
  - Realizar "file scan" para recuperar as tuplas e aplicar seleções-projeções.
  - Escrever as tuplas.
  - Ordenar as tuplas (implementando a cláusula GROUP BY.)

- **Planos utilizando índices**:
  - "Single-Index-Access-Path "
  - "Multiple-Index-Access-Path"
  - "Sort-Index-Access-Path"
  - "Index-Only-Acess-Path"

Estimativas de Custo para Planos de uma Única Relação

- Índice I na chave primária casa com a seleção:
  - *Custo é Altura(I)+1 para uma árvore B+, aproximadamente 1,2 para índice hash.*
- Índice agrupado I casa uma ou mais seleções:
  - *(NPáginas(I)+NPáginas(R)) * produto de FR's das seleções casadas.*
- Índice não-agrupado I casando uma ou mais seleções:
  - *(NPáginas(I)+NTuplas(R)) * produto de FR's das seleções casadas.*
- Varredura seqüencial do arquivo:
  - *NPáginas(R).*

☞ ***Nota:*** *Geralmente, sem eliminação de duplicatas nas projeções! (Exceção: Executada na resposta se o usuário especificou DISTINCT.)*

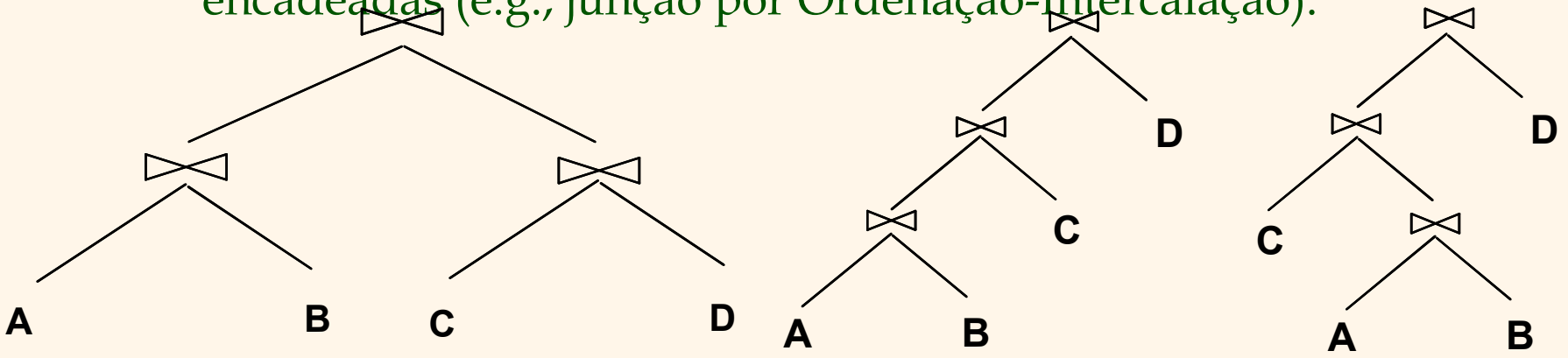
Exemplo

```
SELECT  S.sid
FROM  Sailors S
WHERE  S.rating=8
```

- Se tivermos um índice em *rating*:
  - Cardinalidade: (1/NChaves(I)) * NTuplas(R) = (1/10) * 40000 tuplas
  - Índice agrupado: (1/NChaves(I)) * (NPáginas(I)+NPáginas(R)) = (1/10) * (50+500) páginas são lidas. (Isto é o ***custo***.)
  - Índice não-agrupado: (1/NChaves(I)) * (NPáginas(I)+NTuplas(R)) = (1/10) * (50+40000) páginas são lidas.
- Se tivermos um índice em *sid*:
  - Teria que ler todas as tuplas/páginas. Com um índice agrupado, o custo é 50+500, com índice não-agrupado, 50+40000.
- Fazendo uma varredura de arquivo:
  - Todas as páginas do arquivo são lidas (500).

Consultas sobre Múltiplas Relações

- Decisão fundamental no Sistema R: *somente árvores de junção com profundidade à esquerda* são consideradas.
  - À medida que o número de junções aumenta, o número de planos alternativos cresce rapidamente; *precisamos restringir o espaço de planos.*
  - Árvores de profundidade à esquerda nos permitem gerar todos os planos *totalmente encadeados*.
    - Resultados intermediários não são gravados em arquivos temporários.
    - Nem todas as árvores de profundidade à esquerda são totalmente encadeadas (e.g., junção por Ordenação-Intercalação).



Enumeração de Planos de Profundidade à Esquerda

- Planos de profundidade à esquerda diferem somente na ordem das relações, o método de acesso para cada relação, e o método de junção para cada junção.
- Enumerado usando N passos (se N relações são juntadas):
  - Passo 1: Encontrar o melhor plano de relação única para cada relação.
  - Passo 2: Encontrar a melhor forma de juntar o resultado de cada plano de única relação (como a mais externa) com a outra relação *(Todos os planos de 2 relações.)*
  - Passo N: Encontrar a melhor forma de juntar o resultado de um plano de (N-1) relações (como o mais externo) com a N-ésima relação *(Todos os planos de N relações.)*
- Para cada subconjunto de relações, manter apenas:
  - Plano mais barato no geral, mais
  - Plano mais barato para cada *ordenação interessante* das tuplas.

Enumeração de Planos (Cont.)

- ORDER BY, GROUP BY, agregados etc. manipulados como um passo final, usando tanto um plano `ordenado de forma interessante' ou um operador adicional de ordenação.
- Um plano na etapa N-1 não é combinado com uma relação adicional a não ser que exista uma condição de junção entre eles, a não ser que todos os predicados no WHERE já tenham sido usados.
  - i.e., se possível, evitar produtos cartesianos.
- Apesar de podar o espaço de planos, este enfoque ainda é exponencial no nr. de tabelas.

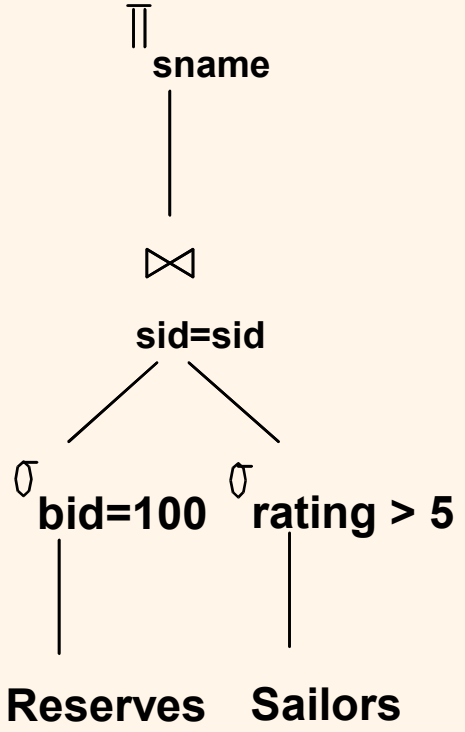
Estimativa de Custo para Planos de Múltiplas Relações

SELECT lista de atributos
FROM lista de relações
WHERE termo1 AND ... AND termok

- Considere um bloco de consulta:
- O nr. máximo de tuplas no resultado é o produto das cardinalidades das relações na cláusula FROM.
- *Fator de redução (FR)* associado a cada *termo* reflete o impacto do *termo* na redução do tamanho do resultado. *Cardinalidade do resultado* = Nr. máximo de tuplas * produto de todos os FR's.
- Planos de múltiplas relações são construídos juntando-se uma nova relação por vez.
  - Custo do método de junção mais estimativa da cardinalidade da junção resultam estimativa de custo e do tamanho do resultado

Exemplo

Sailors:
Árvore B+ em *rating*
Hash em *sid*
Reserves:
Árvore B+ em *bid*



- Passo 1:
  - *Sailors*: Árvore B+ casa *rating>5*, e é provavelmente a mais barata. Todavia, se é esperado que a seleção devolva muitas tuplas, e o índice não é agrupado, então varredura de arquivo pode ser mais barato.
    - Mesmo assim, o plano da árvore B+ é mantido (já que as tuplas estão em ordem de *rating*).
  - *Reserves*: Árvore B+ em *bid* casa *bid=100*; mais barato.
- Passo 2:
  - Consideramos cada plano mantido no Passo 1 como o mais externo, e consideramos como juntá-lo com a (única) outra relação.
    - e.g., *Reserves como mais externa*: índice hash pode ser usado para obter tuplas de Sailors que satisfaçam *sid* = valor de *sid* da tupla mais externa.

Plano Enumerado do Sistema R Exemplo:

```
SELECT S.sname, B.bname, R.day
FROM Sailors S, Reserves R, Boats B
WHERE S.sid = R.sid AND R.bid = B.bid
```

Assumindo:

- *Dois algoritmos de junção:*
  - Sort-Merge-Join
  - Page-Oriented-NL-Join (Incluindo Índice NL-Join)
- *Árvore B + Agrupada em S.sid (Altura = 3 e 500 Páginas de reg. Folha).*
- *S tem 10.000 Páginas, 5 tuplas/pág.*
- *R tem 10 páginas, 10 tuplas/pág.*
- *B tem 10 páginas, 20 tuplas/pág.*
- *10 R ⋈ S ocupando (em média) uma página.*
- *10 R ⋈ B ocupando (em média) uma página.*

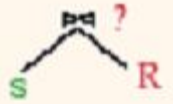
Passo 1: Subplano de uma Relação Simples

- S : (a) Varredura Sequencial ou (b) Varredura Indexada em S.sid.
  - a) Custo de Varredura Sequencial : 10.000
  - b) Custo de Varredura Indexada : 500 + 10.000 = 10.500

Retêm ambos uma vez que (b) mantêm ordenação em sid

- R : Varredura Sequencial somente
  - Custo : 10
- B : Varredura Sequencial somente
  - Custo : 10

Passo 2: Subplano de Relação-2

Iniciando S como sendo bloco externo 

- Varredura Sequencial em S como bloco externo.
  - NL-Join com R : Custo = 10.000 + 10.000(10) = 110.000
  - SM-Join com R : Custo = 10.000 + 2(10.000) + 3(10.000) = 30.030
- Varredura Indexada em S como bloco externo.
  - NL-Join com R : Custo = 10.500 + 10.000(10) = 110.500
  - SM-Join com R : Custo = 10.500 + 3(10) = 10.530

Retêm somente (d)

- Nota : bons planos para S⋈R exploram "ordens interessantes" de subplanos não-ótimos.

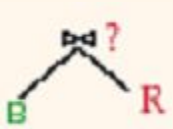
Passo 2 (cont)

Iniciando R como sendo bloco externo

- Junção com S :
  a) NL-Join com S : Custo = 10 + 10.000(10) = 100.010
  b) NL-Join indexado com S indexado : Custo = 10 + 4(100) = 410
  c) SM-Join com S : Custo = 10 + 2(10) + 10.000 + 2(10.000) = 30.030
  d) SM-Join com S indexado : Custo = 10 + 2(10) + 10.500 = 10.530
- Junção com B:
  - NL-Join com B : Custo = 10 + 10(10) = 110
  - SM-Join com B : Custo = 10 + 2(10) + 3(10) = 60

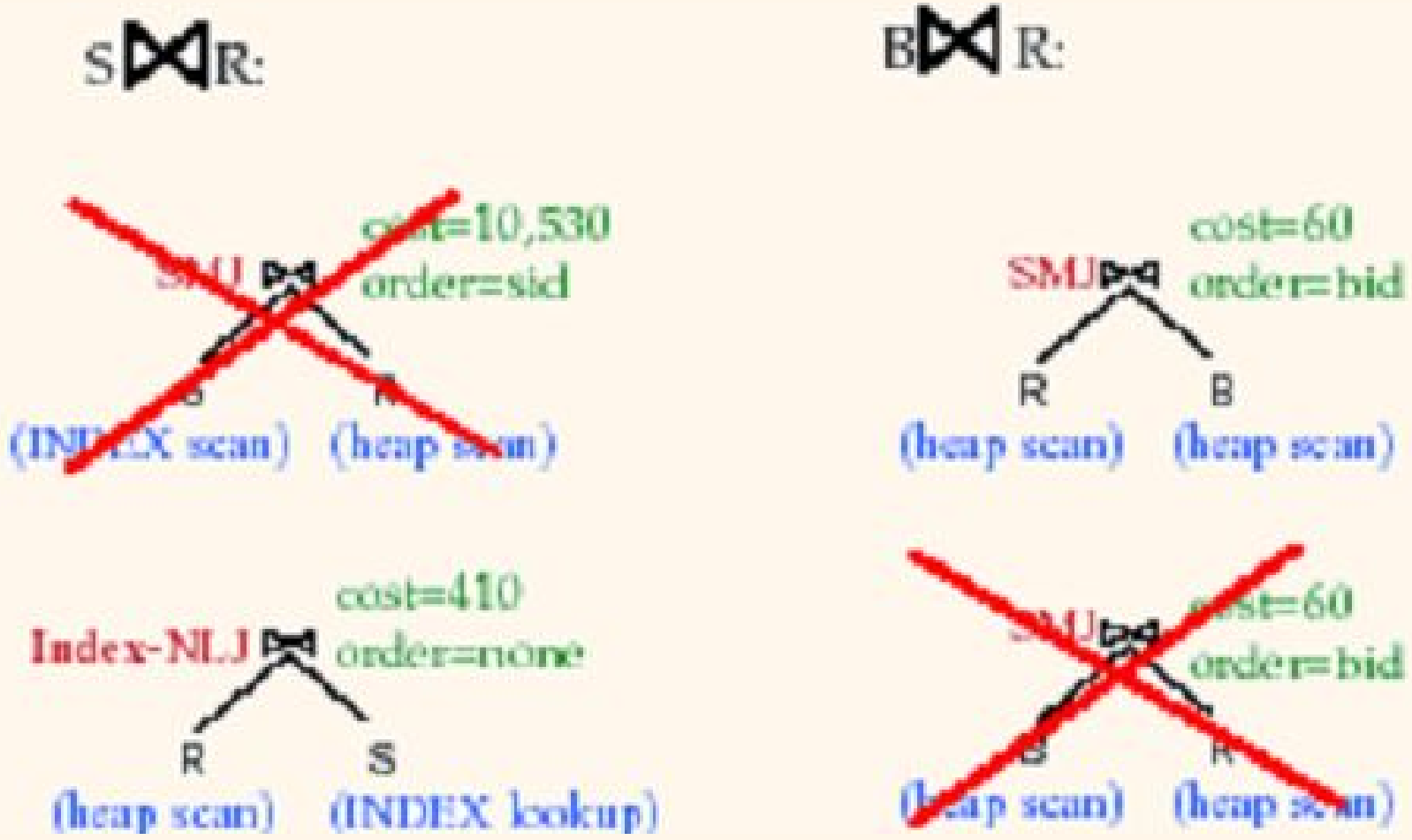
Passo 2 (cont)

Iniciando B como sendo bloco externo

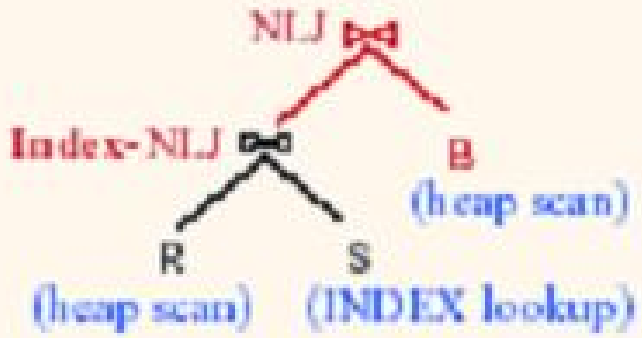


- Junção com B:
  - NL-Join com R : Custo = 10 + 10(10) = 110
  - SM-Join com R : Custo = 10 + 2(10) + 3(10) = 60

Realizando a "poda" de 2-subplanos

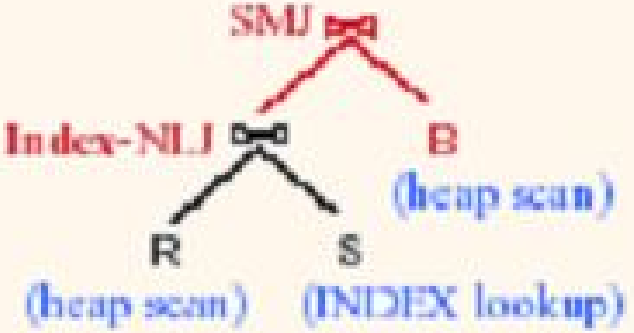


Passo 3 : Subplano da Relação-3



cost = 410 + 10(10) = 510

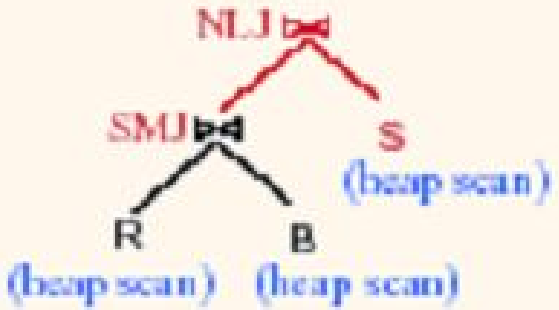
S ⋈ R subplan:
cost=410
order=none
result size = 10 pages



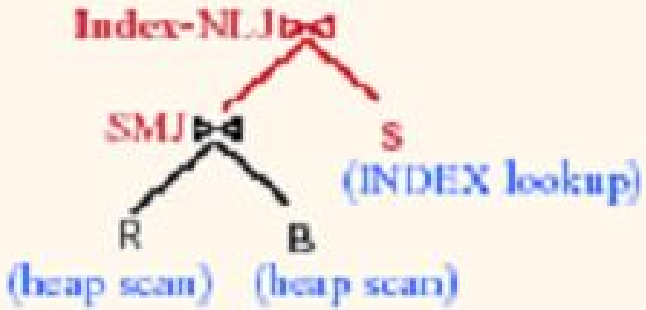
cost = 410 + 2*10 + 3*10 = 460

Passo 3 (cont)

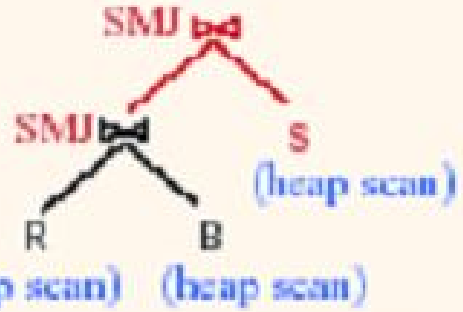
B⋈R subplan:
cost=60, order=bid
result size = 100 tuples (10 pages



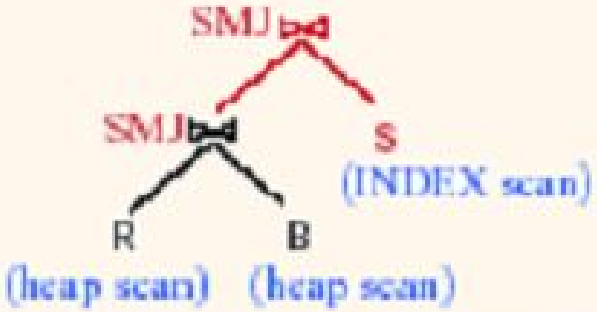
cost = 60 + 10(10,000) = 100,060



cost = 60 + 100*4 = 460

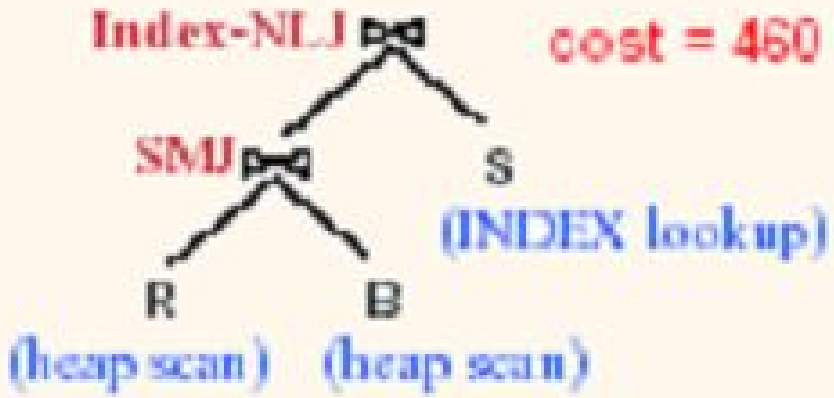


cost = 60 + 10*2 + 3*10,000 = 30,080



cost = 60 + 10*2 + 10,500 = 10,580

E O VENCEDOR É ...



- OBSERVAÇÕES:
  - Bons planos permutam algoritmos de junção.
  - Piores planos têm custo > 10.000
  - Lucro de Otimização ~ 1000 “fold” de aproveitamento em comparação à planos não otimizados.

Consultas Aninhadas

```
SELECT  S.sname
FROM  Sailors S
WHERE EXISTS
   (SELECT  *
    FROM  Reserves R
    WHERE  R.bid=103
    AND  R.sid=S.sid)
```

- O bloco aninhado é otimizado independentemente, com a tupla mais externa considerada como provedora de uma condição de seleção.
- O bloco mais externo é otimizado levando em consideração o custo de `chamar' e calcular o bloco aninhado.
- Ordenação implícita destes blocos significa que algumas boas estratégias não são consideradas. *A versão não-aninhada desta consulta normalmente é melhor otimizada.*

Bloco aninhado a otimizar:

```
SELECT  *
FROM  Reserves R
WHERE  R.bid=103
   AND  R.sid= valor ext.
```

Consulta equival. s/ aninhamento:

```
SELECT  S.sname
FROM Sailors S, Reserves R
WHERE  S.sid=R.sid
  AND R.bid=103
```

Exemplo : Consulta não correlacionada

```
SELECT S.sid
FROM Sailors S
WHERE EXISTS
  (SELECT *
   FROM Reserves R
   WHERE R.bid=103
   AND R.sid=S.sid)
```

Equivalent uncorrelated query:

```
SELECT S.sid
FROM Sailors S
WHERE S.sid IN
  (SELECT R.sid
   FROM Reserves R
   WHERE R.bid=103)
```

Vantagem: Blocos aninhados são executados somente uma vez (e não uma vez por tupla de S).

Exemplo : "Achatando" uma Consulta

```
SELECT S.sid
FROM Sailors S
WHERE S.sid IN
  (SELECT R.sid
   FROM Reserves R
   WHERE R.bid=103)
```

Equivalent non-nested query:

```
SELECT S.sid
FROM Sailors S, Reserves R
WHERE S.sid=R.sid
  AND R.bid=103
```

Vantagem: Uso de algoritmos de junção e otimização aplicada entre algoritmos de junção e reordenação

Sumário

- Otimização de consultas é uma tarefa importante em um SGBD relacional.
- Deve-se entender otimização para entender o impacto de desempenho de um determinado projeto de banco de dados (relações, índices) em uma carga de trabalho (conjunto de consultas).
- Duas partes para otimizar uma consulta:
  - Considerar um conjunto de planos alternativos.
    - Deve-se podar o espaço de planos; tipicamente, somente planos de profundidade à esquerda.
  - Deve-se estimar o custo de cada plano sendo considerado.
    - Deve-se estimar o tamanho do resultado e o custo de cada nó do plano.
    - *Pontos chave*: Estatísticas, índices, implementações de operadores.

Sumário (Continuação)

- Consultas em uma única relação:
  - Todos os caminhos de acesso são considerados, o mais barato é escolhido.
  - *Considerações:* Seleções que *casam* índice, se a chave do índice possui todos os campos necessários e/ou provê tuplas em uma ordem desejada.
- Consultas em múltiplas relações:
  - Todos os planos de uma única relação são enumerados primeiramente.
    - Seleções/projeções consideradas o mais cedo possível.
  - Depois, para cada plano de uma relação, todas as formas de juntar outra relação (como mais interna) são consideradas.
  - Depois, para cada plano de duas relações que é `mantido', todas as formas de juntar outra relação (mais interna) são consideradas etc.
  - Em cada nível, para cada subconjunto de relações, somente o melhor plano para cada ordem interessante de tuplas é `mantido'.