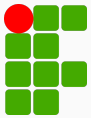
Banco de dados II

05C - Avaliando Operadores Relacionais

Marcos Roberto Ribeiro



Instituto Federal Minas Gerais - Campus Bambuí

2018

Seleção

Esquema considerado:

marinheiros(id_marin: integer, nome_marin: string,
avaliacao: integer, idade: integer)

reservas(id_marin: integer, id_barco: integer,
dia: date, nome_resp: string)

Exemplo

```
SELECT *
FROM reservas
WHERE nome_resp = 'Joe';
```

- Se não houver índice sobre nome_resp, é preciso varrer a toda tabela
- E se houver índice?

Seleção e Índices

Índices de Árvore B+

- Considerando uma seleção $\sigma_{(\mathtt{A}\ \mathbf{op}\ v)}$
- Se o índice for agrupado, é melhor usá-lo
- Senão, é preciso considerar o número de tuplas qualificadas
- No caso de seleções de intervalos, muitas vezes, será melhor varrer toda a tabela do que usar o índice

Índices de Hashing

- Como estes índices suportam apenas seleção por igualdade, sempre devemo usá-los para este tipo de operação

Condições da Seleção

- As seleções podem ter condições mais complexas do que simples comparações
- Estas condições são inicialmente convertidas para a *forma normal conjuntiva*
- Exemplo (Condição original)

```
(dia < '08/09/2017' AND nome_resp = 'Joe')
OR id_barco = 5 OR id_marin = 3
```

- Exemplo (Condição em FNC)

```
(dia < '08/09/2017' OR id_barco = 5 OR id_marin = 3)
AND
(nome_resp = 'Joe' OR id_barco = 5 OR id_marin = 3)
```

Avaliando Seleções sem Disjunção

- Primeira opção:
  1. Calcular o caminho mais seletivo (de acordo com parte da condição)
  2. Obter as tuplas por este caminho e aplicar os demais componentes da condição para obter o resultado final
- Segunda opção (usar vários índices):
  1. Obter os *rids* de todos os índices que satisfazem a condição
  2. Fazer uma interseção entre estes *rids*
  3. Aplicar o restante da condição de seleção

Exemplo

- Condição: `dia < '08/09/2017' AND id_barco = 5 AND id_marin = 3`
- Índice de Árvore B+ sobre `dia`
- Índice de Hashing sobre `id_marin`
  1. **rids**(`dia < '08/09/2017'`) $\cap$ **rids**(`id_marin = 3`)
  2. Filtragem das tuplas com `id_barco = 5`

Seleções com Disjunções

- Um único termo da disjunção pode provocar uma varredura de arquivo

Exemplo

- Supondo que haja somente os índices:
  - Hashing sobre `nome_resp`
  - Hashing sobre `id_margin`
- Condição: `(dia < '08/09/2017') OR (nome_resp = 'Joe')`
  - A condição `(dia < '08/09/2017')` causa uma varredura
  - Uma alternativa é realizar a varredura e testar se o registro satisfaz uma das duas condições
- Condição: `((dia < '08/09/2017') OR (nome_resp = 'Joe') AND id_marin = 3)`
  - Neste caso o índice sobre `id_marin` pode ser usado para selecionar as tuplas qualificadas e, em seguinda, as demais condições podem ser verificadas

Projeção

```
SELECT DISTINCT id_marin, id_barco
FROM reservas;
```

Avaliação de Projeções

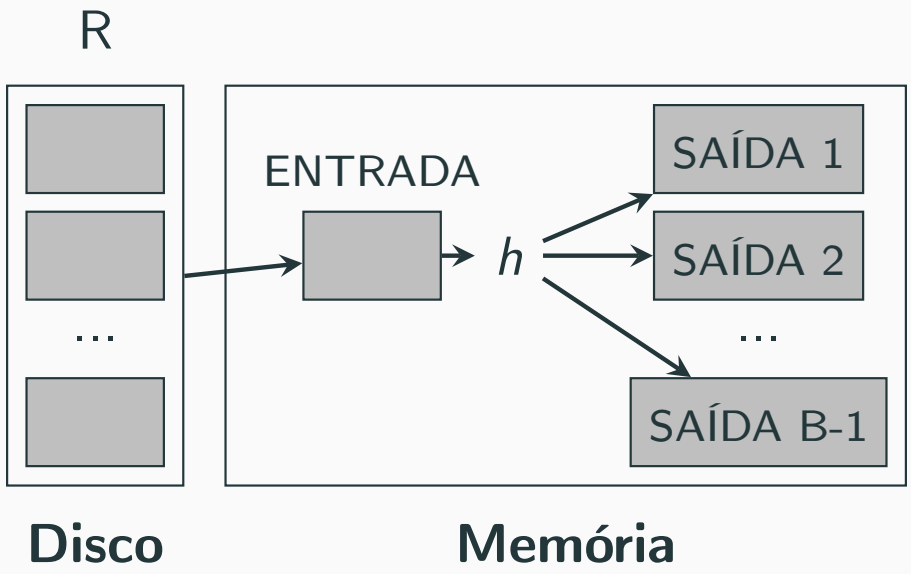
1. Remoção dos atributos indesejados
2. Eliminação das tuplas duplicadas

- O segundo passo é o mais difícil
- Duas técnicas principais podem ser usadas:
  - Projeção Baseada em Ordenação
  - Projeção Baseada em Hashing

Projeção Baseada em Ordenação

- A projeção baseada em ordenação é feita da seguinte maneira:
  1. Percorrer R e produzir um conjunto de tuplas com os atributos desejados (Custo: $M + T$ E/S, onde M é o número de páginas de R e T é o número de páginas do conjunto produzido)
  2. Ordenar esse conjunto de tuplas (Custo: $O(T \log T$ E/S, como $T = O(M)$, temos $M \log M$ E/S)
  3. Percorrer o resultado ordenado, comparando tuplas adjacentes, e descartar as duplicatas (Custo: $O(T) = M$ E/S)
- Custo total: $2M + M \log M$ E/S
- Uma otimização interessante é eliminar as duplicatas na fase de intercalação da ordenação

Projeção Baseada em Hashing: Particionamento



Fase 1: Particionamento

- Lemos uma página de R por vez
- Para cada tupla da página
  - Aplicamos uma função hash h sobre os atributos projetados
  - Gravamos a tupla na partição correspondente a h
- Páginas de saída cheias são gravadas em disco (uma partição pode ter várias páginas)
- É garantido que tuplas de partições diferentes não são duplicatas

Projeção Baseada em Hashing: Eliminação de Duplicatas

- Crie uma tabela hash em memória
- Leia cada partição da face anterior (uma página por vez)
- Para cada tupla da partição
  - Aplique a função h' (diferente de h) e armazene a tupla na tabela hash
  - Elimine duplicatas para tuplas com mesmo valor de h'
  - Após a leitura completa da partição, grave o resultado em arquivo
- Se a tabela hash não couber em memória, podemos aplicar o particionamento recursivamente

Custo

- Supondo R armazenada em M páginas e a projeção de R (sem eliminar duplicatas) armazenada em T páginas
- Cada partição contém $O(\frac{T}{B-1}) = \frac{M}{B-1}$ páginas
- Particionamento: $M + T$ E/S
- Eliminação de duplicatas: T E/S
- Custo Total: $O(M + 2T) = 3M$ E/S

Projeção

Ordenação x Hashing

- Em geral, a ordenação é superior
  - Principalmente quando há muitas duplicatas
- A ordenação retorna um resultado ordenado
- Os SGBD já possuem a ordenação externa implementada

Uso de Índice para Projeções

- Um índice pode ser útil se a chave de pesquisa contém todos os atributos da projeção
- Podemos recuperar os valores das chaves e aplicar uma das técnicas de projeção
- Não acessamos a relação, apenas o índice que é bem menor
- No caso dos atributos da projeção serem o prefixo da chave de pesquisa de uma árvore B+, a projeção pode ser mais eficiente ainda. Por quê?

Junção

- A junção é uma operação muito importante e largamente usada no processamento de consultas
- Exemplo:

```
SELECT *
FROM reservas r, marinheiros m
WHERE r.id_marin = m.id_marin
```

- A consulta anterior pode ser representada em álgebra relacional pela expressão $r \bowtie_{\texttt{id\_marin}} m$
- As junções podem ser representadas por operações com produtos cartesianos, seleções e projeções
- Entretanto, o uso de algoritmos específicos para o processamento de junções é mais eficiente do que o processamento destas operações equivalentes

Junção de Loops Aninhados

Algoritmo para Processar a Junção $S \bowtie_A S$

```
1 para cada tupla t ∈ R faça
2     para cada tupla t' ∈ S faça
3         se R.A = S.A então
4             Adiciona (t, t') ao resultado;
```

- Supondo R com M páginas e p_R tuplas / página, S com N páginas
- O custo do algoritmo é $M + M \times p_R \times N$ E/S
- Podemos melhorar o algoritmo, se lermos uma página de cada tabela e fizermos a junção das tuplas destas páginas
- Neste caso, o custo cai para $M \times N$ E/S

Junção de Loops Aninhados de Bloco

- O algoritmo de junção de loops aninhados simples não utiliza o buffer de memória de forma eficiente
- Supondo que hajam B páginas de memória livre
  - Podemos ler $B - 2$ páginas da relação externa
  - Percorremos a relação interna usando uma página
  - E usamos a última página para gravar o resultado
  - O custo será de $M + N \times \frac{M}{B-2}$ E/S

Algoritmo

```
1 para cada bloco b_r de B − 2 páginas de R faça
2     para cada página p_S de S faça
3         para cada tupla t ∈ b_r faça
4             para cada tupla t' ∈ p_S faça
5                 se R.A = S.A então
6                     Adiciona (t, t') ao resultado;
```

Junção de Loops Aninhados Indexados

- Se houver um índice em uma das relações sobre o(s) atributo(s) da junção, podemos tirar proveito do índice e tornar esta relação a "interna"
- O algoritmo lê as tuplas da relação externa e compara apenas com o índice da relação interna
- Como o índice ocupa menos espaço, podemos ler mais tuplas da relação interna de uma vez
- Custo: $M \times N$ E/S (este é o pior caso, se houverem poucas correspondências o custo será bem menor)

Junção Sort-Merge

- As duas relações são ordenadas sobre o atributo de junção
- A ordenação agrupa as tuplas com o mesmo valor com o mesmo valor no atributo de junção em partições
- Explorando este particionamento, comparamos as tuplas de R apenas com as tuplas de S que estão na mesma partição
- Após a ordenação, começamos na primeira tupla de cada relação e avançamos quando o valor for menor do que da outra relação

Exemplo de Junção Sort-Merge

Marinheiros

id_marin	nome_marin	avaliacao	idade
22	dustin	7	45,0
28	yuppy	9	35,0
31	lubber	8	55,5
36	lubber	6	36,0
44	guppy	5	35,0
58	rusty	10	35,0

Reservas

id_marin	id_barco	dia	nome_resp
28	103	12/04/1996	guppy
28	103	11/03/1996	yuppy
31	101	10/10/1996	dustin
31	102	10/12/1996	lubber
31	101	10/11/1996	lubber
58	103	11/12/1996	dustin

Custo

- Normalmente é próximo de $M \log M + N \log N + M + N$ E/S
- Porém, se houver correspondências em todas as tuplas, será de $M \log M + N \log N + M \times N$ E/S

Exemplo de Junção Sort-Merge

Marinheiros

id_marin	nome_marin	avaliacao	idade
22	dustin	7	45,0
28	yuppy	9	35,0
31	lubber	8	55,5
36	lubber	6	36,0
44	guppy	5	35,0
58	rusty	10	35,0

Reservas

id_marin	id_barco	dia	nome_resp
28	103	12/04/1996	guppy
28	103	11/03/1996	yuppy
31	101	10/10/1996	dustin
31	102	10/12/1996	lubber
31	101	10/11/1996	lubber
58	103	11/12/1996	dustin

Custo

- Normalmente é próximo de $M \log M + N \log N + M + N$ E/S
- Porém, se houver correspondências em todas as tuplas, será de $M \log M + N \log N + M \times N$ E/S

Exemplo de Junção Sort-Merge

Marinheiros

id_marin	nome_marin	avaliacao	idade
22	dustin	7	45,0
28	yuppy	9	35,0
31	lubber	8	55,5
36	lubber	6	36,0
44	guppy	5	35,0
58	rusty	10	35,0

Reservas

id_marin	id_barco	dia	nome_resp
28	103	12/04/1996	guppy
28	103	11/03/1996	yuppy
31	101	10/10/1996	dustin
31	102	10/12/1996	lubber
31	101	10/11/1996	lubber
58	103	11/12/1996	dustin

Custo

- Normalmente é próximo de $M \log M + N \log N + M + N$ E/S
- Porém, se houver correspondências em todas as tuplas, será de $M \log M + N \log N + M \times N$ E/S

Exemplo de Junção Sort-Merge

Marinheiros

id_marin	nome_marin	avaliacao	idade
22	dustin	7	45,0
28	yuppy	9	35,0
31	lubber	8	55,5
36	lubber	6	36,0
44	guppy	5	35,0
58	rusty	10	35,0

Reservas

id_marin	id_barco	dia	nome_resp
28	103	12/04/1996	guppy
28	103	11/03/1996	yuppy
31	101	10/10/1996	dustin
31	102	10/12/1996	lubber
31	101	10/11/1996	lubber
58	103	11/12/1996	dustin

Custo

- Normalmente é próximo de $M \log M + N \log N + M + N$ E/S
- Porém, se houver correspondências em todas as tuplas, será de $M \log M + N \log N + M \times N$ E/S

Exemplo de Junção Sort-Merge

Marinheiros

id_marin	nome_marin	avaliacao	idade
22	dustin	7	45,0
28	yuppy	9	35,0
31	lubber	8	55,5
36	lubber	6	36,0
44	guppy	5	35,0
58	rusty	10	35,0

Reservas

id_marin	id_barco	dia	nome_resp
28	103	12/04/1996	guppy
28	103	11/03/1996	yuppy
31	101	10/10/1996	dustin
31	102	10/12/1996	lubber
31	101	10/11/1996	lubber
58	103	11/12/1996	dustin

Custo

- Normalmente é próximo de $M \log M + N \log N + M + N$ E/S
- Porém, se houver correspondências em todas as tuplas, será de $M \log M + N \log N + M \times N$ E/S

Exemplo de Junção Sort-Merge

Marinheiros

id_marin	nome_marin	avaliacao	idade
22	dustin	7	45,0
28	yuppy	9	35,0
31	lubber	8	55,5
36	lubber	6	36,0
44	guppy	5	35,0
58	rusty	10	35,0

Reservas

id_marin	id_barco	dia	nome_resp
28	103	12/04/1996	guppy
28	103	11/03/1996	yuppy
31	101	10/10/1996	dustin
31	102	10/12/1996	lubber
31	101	10/11/1996	lubber
58	103	11/12/1996	dustin

Custo

- Normalmente é próximo de $M \log M + N \log N + M + N$ E/S
- Porém, se houver correspondências em todas as tuplas, será de $M \log M + N \log N + M \times N$ E/S

Exemplo de Junção Sort-Merge

Marinheiros

id_marin	nome_marin	avaliacao	idade
22	dustin	7	45,0
28	yuppy	9	35,0
31	lubber	8	55,5
36	lubber	6	36,0
44	guppy	5	35,0
58	rusty	10	35,0

Reservas

id_marin	id_barco	dia	nome_resp
28	103	12/04/1996	guppy
28	103	11/03/1996	yuppy
31	101	10/10/1996	dustin
31	102	10/12/1996	lubber
31	101	10/11/1996	lubber
58	103	11/12/1996	dustin

Custo

- Normalmente é próximo de $M \log M + N \log N + M + N$ E/S
- Porém, se houver correspondências em todas as tuplas, será de $M \log M + N \log N + M \times N$ E/S

Exemplo de Junção Sort-Merge

Marinheiros

id_marin	nome_marin	avaliacao	idade
22	dustin	7	45,0
28	yuppy	9	35,0
31	lubber	8	55,5
36	lubber	6	36,0
44	guppy	5	35,0
58	rusty	10	35,0

Reservas

id_marin	id_barco	dia	nome_resp
28	103	12/04/1996	guppy
28	103	11/03/1996	yuppy
31	101	10/10/1996	dustin
31	102	10/12/1996	lubber
31	101	10/11/1996	lubber
58	103	11/12/1996	dustin

Custo

- Normalmente é próximo de $M \log M + N \log N + M + N$ E/S
- Porém, se houver correspondências em todas as tuplas, será de $M \log M + N \log N + M \times N$ E/S

Exemplo de Junção Sort-Merge

Marinheiros

id_marin	nome_marin	avaliacao	idade
22	dustin	7	45,0
28	yuppy	9	35,0
31	lubber	8	55,5
36	lubber	6	36,0
44	guppy	5	35,0
58	rusty	10	35,0

Reservas

id_marin	id_barco	dia	nome_resp
28	103	12/04/1996	guppy
28	103	11/03/1996	yuppy
31	101	10/10/1996	dustin
31	102	10/12/1996	lubber
31	101	10/11/1996	lubber
58	103	11/12/1996	dustin

Custo

- Normalmente é próximo de $M \log M + N \log N + M + N$ E/S
- Porém, se houver correspondências em todas as tuplas, será de $M \log M + N \log N + M \times N$ E/S

Junção por Hashing

- Possui duas fases:
  1) **Particionamento:** Fazemos hashing das duas relações no atributo de junção, usando a mesma função de hashing h
  2) **Correspondência:** Comparamos as tuplas em uma partição de R somente com as tuplas na partição correspondente de S
- Na fase de correspondência, para cada partição de R, construímos uma tabela hash em memória usando uma função h' (diferente de h)
- Lemos então, a partição de S correspondente e comparamos apenas as tuplas com mesmo valor de h'
- Custo: $3(M + N)$ E/S (este custo pode ser maior se não houver memória para conter uma partição)

Operações de Conjunto

- A operação $R \cap S$ pode ser vista como um caso especial de junção
  - Junção com igualdade em todos os atributos $R \bowtie_{A_1=A'_1 \wedge \ldots \wedge A_n=A'_n} S$
- União e diferença (com eliminação de duplicatas)
  - Ordenação
  - Hashing

União e Diferença Usando Ordenação

1. Ordene R e S considerando todos os atributos

2. Percorra R e S ordenadas em paralelo e intercale, eliminando as duplicatas

   União ($R \cup S$): pegue cada tupla t presente em R e S

   Diferença ($R - S$): pegue $t \in R$ tal que $t \in S$

União e Diferença Usando Hashing

1. Particione R e S usando uma função de hashing h
2. Para cada partição l
   2.1 Construa uma tabela de hashing H em memória para R_l (usando uma função $h' \neq h$)
   2.2 Para cada tupla $t \in S_l$

   União ($R \cup S$): Se $t \notin H$, insira t em H

   Diferença ($R - S$): Se $t \in H$, remova t de H
3. Adicione as tuplas de H ao resultado

Operações Agregadas

- Algoritmo básico
  - Varre a relação
  - Mantém informações em memória

Operação	Informação
SUM	total
AVG	total, contagem
COUNT	contagem
MIN	menor valor
MAX	maior valor

- Cláusula **GROUP BY**
  - Agrupa o resultado da função

Operações Agregadas: Estratégias

Estratégia de Ordenação

- Ordena a relação pelos atributos de agrupamento
- Percorre a relação novamente para calcular a operação agregada
- Possui o mesmo custo da ordenação: $O(M \log M)$

Estratégia de Hashing

- Constrói uma tabela hash
  - valor-agrupamento $\rightarrow$ informação
- Percorremos a relação e atualizamos a tabela hash
- Custo: $O(M)$
- O custo pode ser pior se a tabela hash não couber em memória

Usando Índices

- Pode ser feita uma avaliação somente de índice se houver um índice adequado

Referências I


Date, C. J. (2004).
***Introdução a sistemas de bancos de dados*.**
Elsevier, Rio de Janeiro.


Elmasri, R. and Navathe, S. B. (2011).
***Sistemas de banco de dados*.**
Pearson Addison Wesley, São Paulo, 6 edition.


Ramakrishnan, R. and Gehrke, J. (2008).
***Sistemas de gerenciamento de banco de dados*.**
McGrawHill, São Paulo, 3 edition.


Silberschatz, A., Korth, H. F., and Sudarshan, S. (2007).
Sistema de bancos de dados.