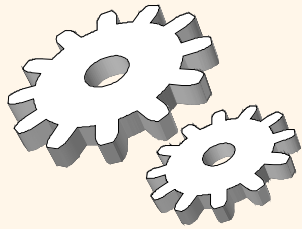
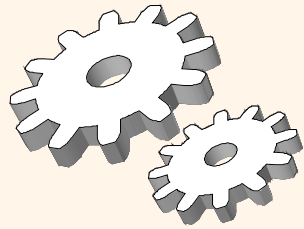


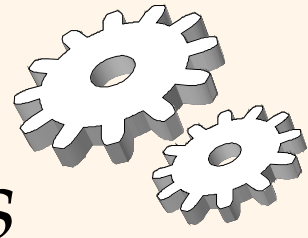
Avaliação de Operações Relacionais: Outras Técnicas

Capítulo 14, Parte B

Usando um Índice para Seleções



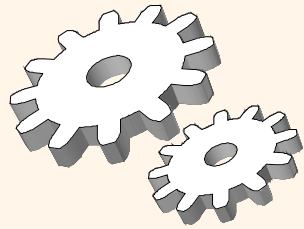
- Custo depende do nº de tuplas qualificadas e agrupamento.
  - Custo de encontrar entradas de dados qualificadas (tipicamente pequeno) mais o custo de ler os registros (pode ser grande sem agrupamento).
  - No exemplo, assumindo distribuição uniforme de nomes, cerca de 10% das tuplas se qualificam (100 páginas, 10000 tuplas). Com um índice agrupado, o custo é pouco mais do que 100 E/Ss; se não-agrupado, até 10000 E/Ss!
- *Refinamento importante para índices não-agrupados*:
  1. Encontrar entradas de dados qualificadas.
  2. Ordenar os rid's dos registros de dados a serem lidos.
  3. Ler rids em ordem. Isto garante que cada página de dados é verificada uma única vez (apesar de o nº de tais páginas ser provavelmente maior do que com agrupamento).



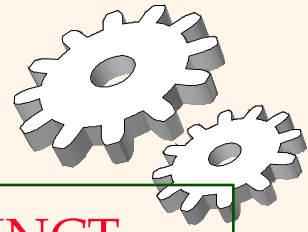
Dois Enfoques para Seleções Gerais

- Primeiro enfoque: Encontrar o *caminho de acesso mais seletivo*, usá-lo para obter as tuplas, e aplicar quaisquer termos restantes que não casam o índice:
  - *Caminho de acesso mais seletivo:* Um índice ou varredura de arquivo que estimamos que irá precisar do menor número de E/Ss de páginas.
  - Termos que casam este índice reduzem o número de tuplas *lidas*; outros termos são usados para descartar algumas tuplas lidas, mas não afetam o número de tuplas / páginas lidas.
  - Considere *day<8/9/94 AND bid=5 AND sid=3*. Um índice de árvore B+ em *day* pode ser usado; então, *bid=5* e *sid=3* devem ser verificados para cada tupla lida. Similarmente, um índice hash em <*bid, sid*> poderia ser usado; então, *day<8/9/94* deve ser verificado.

Interseção de Rids



- Segundo enfoque (se temos 2 ou mais índices que casam e que usam as Alternativas (2) ou (3) para as entradas de dados):
  - Obtenha conjuntos de rids de registros de dados usando cada índice que casa.
  - Então, faça a *interseção* destes conjuntos de rids (logo veremos interseção!)
  - Leia os registros e aplique os termos restantes.
  - Considere *day<8/9/94 AND bid=5 AND sid=3*. Se temos um índice de árvore B+ em *day* e um índice em *sid*, ambos usando a Alternativa (2), podemos ler rids de registros satisfazendo *day<8/9/94* usando o primeiro, rids de registros satisfazendo *sid=3* usando o segundo, fazer a interseção, ler os registros e verificar *bid=5*.

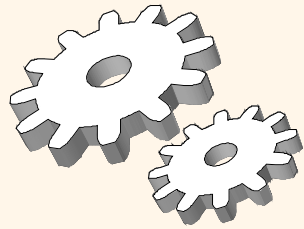


A Operação de Projeção

```
SELECT   DISTINCT
         R.sid, R.bid
FROM     Reserves R
```

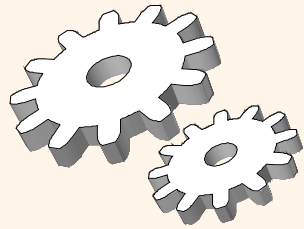
- Um enfoque baseado em ordenação:
  - Modifique o Passo 0 de ordenação externa para eliminar campos não desejados. Portanto, *runs* de cerca de 2B páginas são produzidas, mas tuplas nas rodadas são menores do que as tuplas de entrada. (A razão de tamanho depende do nº e tamanho dos campos que estão sendo eliminados.)
  - Modifique os passos de intercalação para que eliminem duplicatas. Portanto, o número de tuplas resultantes é menor do que a entrada. (A diferença depende do número de duplicatas.)
  - Custo: No Passo 0, ler a relação original (tamanho M), escrever o mesmo número de tuplas menores. Nos passos de intercalação, um número menor de tuplas gravadas em cada passo. Usando o exemplo de Reserves, 1000 páginas de entrada reduzidas a 250 no Passo 0 se a razão de tamanho é 0,25.

Projeção Baseada em Hashing



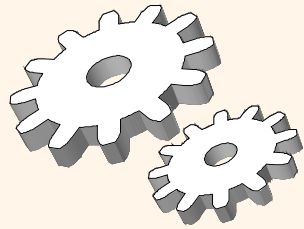
- *Fase de particionamento*: Ler R usando um buffer de entrada. Para cada tupla, descartar campos não desejados, aplicar a função de hash *h1* para escolher um dos B-1 buffers de saída.
  - O resultado são B-1 partições (de tuplas sem campos não desejados). 2 tuplas de partições diferentes são garantidamente distintas.
- *Fase de eliminação de duplicatas*: Para cada partição, lê-la e construir uma tabela hash na memória, usando uma função de hash *h2* (<> *h1*) em todos os campos, descartando as duplicatas.
  - Se a partição não cabe na memória, pode-se aplicar recursivamente um algoritmo de projeção baseado em hash a esta partição.
- Custo: Para o particionamento, ler R, gravar cada tupla, mas com menos campos. Este resultado é lido na próxima fase.

Discussão da Projeção



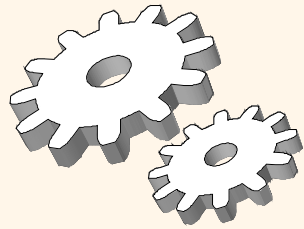
- Enfoque baseado em ordenação é o padrão; melhor processamento da distribuição irregular e o resultado é ordenado.
- Se um índice na relação contém todos os atributos desejados em sua chave de pesquisa, pode-se varrer *apenas o índice*.
  - Aplicar técnicas de projeção nas entradas de dados (muito menores!)
- Se um índice ordenado (i.e., árvore) contém todos os atributos desejados como *prefixo* da chave de pesquisa, pode-se fazer ainda melhor:
  - Ler entradas de dados em ordem (varredura apenas no índice), descartar campos não desejados, comparar tuplas adjacentes para verificar ocorrências de duplicatas.

Operações de Conjuntos

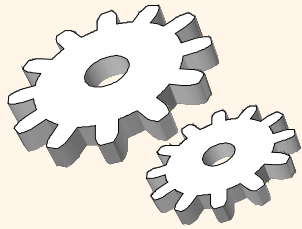


- Interseção e produto cartesiano são casos particulares de junção.
- União (Distinct) e Exceto são similares; faremos a união.
- Enfoque baseado em classificação para a união:
  - Classificar ambas as relações (pela combinação de todos os atributos).
  - Varrer as relações classificadas e intercalá-las.
  - *Alternativa*: A intercalação roda desde o Passo 0 para *ambas* as relações.
- Enfoque baseado em hash para a união :
  - Dividir R e S usando a função de hash *h*.
  - Para cada partição-S, construir uma tabela hash em memória (usando *h2*), varrer a partição-R correlata e adicionar tuplas à tabela, descartando duplicatas.

Operações de Agregados (AVG, MIN etc.)

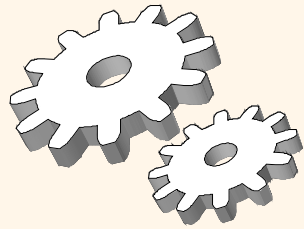


- Sem agrupamento:
  - Geralmente, requer varredura da relação.
  - Com um índice cuja chave de pesquisa inclua todos os atributos nas cláusulas SELECT ou WHERE, pode-se varrer apenas o índice.
- Com agrupamento:
  - Classificar nos atributos do group-by, então varrer a relação e calcular os agregados para cada grupo. (Pode-se melhorar através da combinação de classificação e cálculo dos agregados.)
  - Enfoque similar baseado em cálculo do hash dos atributos do group-by.
  - Com um índice em árvore cuja chave de pesquisa inclua todos os atributos nas cláusulas SELECT, WHERE e GROUP BY, pode-se varrer apenas o índice; se os atributos do group-by formam um prefixo da chave de pesquisa, pode-se ler entradas de dados / tuplas na ordem do group-by.



Impacto da Bufferização

- Se várias operações estão executando concorrentemente, estimar o número de páginas de buffer disponíveis é um trabalho de adivinhação.
- Padrões de acesso repetidos interagem com a política de gerenciamento do buffer.
  - e.g., Relação mais interna é varrida repetidamente em uma Junção por Laço Aninhado Simples. Com páginas de buffer suficientes para manter a mais interna, a política de gerenciamento não importa. Caso contrário, MRU é melhor, LRU é pior (*enxurrada seqüencial*).
  - A política de gerenciamento importa para Laços Aninhados em Bloco?
  - E Laços Aninhados com Índice? Junção por Ordenação-Intercalação?



Sumário

- Uma virtude de SGBDs relacionais: *consultas são compostas por poucos operadores básicos*; a implementação destes operadores deve ser cuidadosamente ajustada (e é importante fazê-lo!).
- Muitas técnicas alternativas de implementação para cada operador; nenhuma técnica universalmente superior para a maior parte dos operadores.
- Deve-se considerar alternativas disponíveis para cada operação em uma consulta e escolher a melhor baseada em estatísticas do sistema etc. Isto é parte da tarefa mais abrangente de otimizar uma consulta composta por várias operações.