DI- FCT/UNL **10 de Janeiro de 2011**

Sistemas de Bases de Dados

Exame de Época Normal, 2010/11

Duração: 3 horas (sem consulta)

Grupo 1

Considere parte duma base de dados de cadastros de criminalidade, incluindo perfis de ADN. Essa base de dados inclui as seguintes tabelas (onde em cada uma delas apenas se apresentam parte dos atributos, e os atributos que constituem a chave primária estão sublinhados):

cidadãos({NumBI,Nome,Sx,NIF,HabLiterarias...}) ocorrências({Ocorr,BICr,BIVitm,Data,Crime...})
cadastrados({NumBI,PerfilADN,DataCad,Perigo}) condena(Proc,Juiz,Ocorr,Data,BICr,Pena,...)
crimes(Crime,Desc,MolduraPenal,...)

Para cada uma destas tabelas existe um índice clustered de árvore B+ sobre o(s) atributo(s) da chave primária.

O sistema de gestão de bases de dados usa blocos de 4KB e, para os exercícios, considere que a memória comporta apenas 100 blocos. Os registos de todas as tabelas têm dimensão variável, sendo que, em média, um registo da tabela de cidadãos ocupa 1KB, um registo de ocorrências ocupa 100 bytes um registo da tabela de condenações (condena) ocupa 200 bytes, um registo da tabela de crimes ocupa 50 bytes. Um registo da tabela de cadastrados ocupa 1MB, mas se não considerarmos o atributo de PerfilADN, ocupa apenas 20 bytes.

Sabemos ainda que num dado momento a tabela de cidadãos tem 10.000.000 tuplos, a de cadastrados 100.000 tuplos, a de ocorrencias 1.000.000 tuplos, a de condenações 500.000 tuplos, e a de crimes tem 1000 tuplos.

Nota: Neste grupo, sempre que se solicitarem exemplos, estes devem ser **exclusivamente** sobre esta base de dados. Além disso, **todas** as respostas deverão conter uma **breve justificação**.

1 a) Apresente dois planos para execução da seguinte pergunta SQL (que retorna os tipos de crimes alegadamente cometidos por pessoas com nome "José"), justificando qual deles deverá ter um menor custo na base de dados em causa.

```
select crimes.Descr
from cidadãos, cadastrados, ocorrências, crimes
where Nome like "%José%" and cidadãos.NumBI = cadastrados.NumBI and
   ocorrências.BICr = cadastrados.NumBI and ocorrências.crime=crimes.crime
```

1 b) Como acha que um SGBD, dos que estudou na disciplina, armazenaria a tabela de cadastrados?

1 c) Considere o seguinte esquema de pergunta SQL, onde ***A*** é um atributo da tabela de cidadãos diferente da chave (i.e. não pode ser NumBI) sobre o qual está definido um ficheiro de índice non-clustered de árvore B+ e ***valor*** é um valor do domínio de ***A***:

select * from cidadãos where *A* = *valor*;

Apresente um exemplo de atributo ***A*** e ***valor*** para o qual a pergunta certamente não usaria o ficheiro de índice, e um outro em que certamente usaria o índice.

1 d) Considere que o sistema apenas implementa para junções o algoritmo de "nested loop join", eventualmente usando índices quando disponíveis.
Para a junção das tabelas de 1a) (i.e. cidadãos⋈cadastrados⋈ocorrências⋈crimes) indique qual lhe parece ser a melhor ordem para fazer as junções e, em cada operação, qual deveria ser a "inner table" e qual deveria ser a "outer table" no "nested loop".

1 e) Dê um exemplo de uma junção de duas das tabelas da bases de dados acima em que o "merge-join" seja o mais adequado. (**Nota:** a pergunta é **apenas** a junção de duas das tabelas acima).

1 f) Apresente um exemplo de plano de execução e pergunta SQL em que não seja possível usar exclusivamente "pipelining" na avaliação de expressões intermédias.

1 g) Considere as seguintes transações concorrentes:

begin transaction	**begin transaction**
select * from crimes;	**select * from crimes;**
insert into crimes values ('a','b',...);	**insert into crimes values ('b','c',...);**
select * from crimes;	**select * from crimes;**
commit;	**commit;**

Apresente um escalonamento das operações nestas duas transações em que o resultado seja diferente consoante se esteja em modo de isolamento Read uncommited, Read commited ou Serializable, mostrando qual o resultado para cada um dos 3 modos de isolamento.

1 h) Para garantir o isolamento em bases de dados não é suficiente fazer locks apenas sobre os tuplos diretamente afectados por cada uma das operações envolvidas numa transação. Apresente um escalonamento de **operações SQL sobre a base de dados acima** em duas transações concorrentes em que se possa constatar isso mesmo, i.e. em que o resultado difira caso se faça apenas lock aos tuplos diretamente envolvidos, ou a toda a tabela envolvida, sendo que só este último caso garante o isolamento.

Grupo 2

Nota: A resposta a cada uma das alíneas deste grupo **não pode em caso algum exceder uma página**.

2 a) *"O algoritmo "external sort-merge" prevê uma primeira passagem pela relação a ordenar, onde são ordenados de forma separada vários "runs" da relação, seguida de uma **ou mais** passagens onde se juntam ("merge") os vários runs. No entanto, na prática é muitíssimo raro ser necessária mais do que uma passagem de junção, pelo que na implementação de alguns SGBDs essa hipótese nem sequer é prevista."*

Justifique que de facto, na prática, esses casos são raros (e.g. mostrando, em casos concretos, a partir de que dimensões de relações é necessária mais do que uma passagem de junção).

2 b) *"A execução concorrente de transações é mais importante quando o acesso a dados é lento e as transações são longas, e menos importante quando o acesso a dados é rápido e as transações são curtas."*

Justifique porque razões tal se passa.

2 c) Há essencialmente duas formas distintas de distribuir uma base de dados: por replicação e por fragmentação. Em que é que estas duas formas de distribuição diferem? Em qual destas formas de distribuição é útil a estratégia de "semi-join" e porquê?

.