Universidade Federal do Espírito Santo
Centro Tecnológico
Departamento de Informática

Banco de Dados (INF15979)
Prof. Rodrigo Laiola Guimarães

Especificação do Trabalho

O segundo trabalho da disciplina de Banco de Dados I consiste na implementação de um algoritmo para verificação se um escalonamento (*schedule*) é estrito. Este trabalho tem por objetivo auxiliar os(as) alunos(as) na compreensão da teoria de processamento de transações, em particular, na caracterização de *schedules* com base na sua facilidade de recuperação. Cada grupo pode ser composto por até 2 alunos(as) e deve utilizar um dos sistemas de gerenciamento de banco de dados utilizados na disciplina: MySQL versão 5.7 ou PostgreSQL 10.

Teste de *schedule* estrito
Em um *schedule* estrito as transações não podem ler nem gravar um item X até que a última transação que gravou X tenha sido confirmada (ou cancelada). *Schedules* estritos simplificam o processo de recuperação. Em um *schedule* estrito, o processo de desfazer uma operação write_item(X) de uma transação abortada serve apenas para restaurar a imagem anterior (valor_antigo ou BFIM) do item de dados X. Esse procedimento simples sempre funciona corretamente para *schedules* estritos, mas pode não funcionar para *schedules* recuperáveis ou sem cascata. Como ilustração, considere os 2 *schedules* mostrados abaixo, que realizam algumas operações. No Exemplo 1, T2 lê o valor de Y (`time=10`) depois que T3 o modifica e é confirmada (`time=8` e `time=9`, respectivamente). Já no Exemplo 2, T2 lê Y (`time=37`) antes que T3, última transação a modificar o item (`time=35`), tenha sido confirmada (`time=48`). O algoritmo de teste de *schedule* estrito deve possuir a assinatura descrita nos arquivos em anexo.

Entrada:
Considere a existência da tabela **`Schedule`**, na qual cada linha representa a chegada de uma operação pertencente à transação. A tabela possui 4 colunas: a primeira representa o tempo de chegada (`time`), a segunda o identificador da transação (`#t`), a terceira a operação (`R` : leitura, `W` : escrita, `C` : confirmação ou `A` : aborto) e a quarta o atributo que será lido/escrito (quando aplicável). As linhas da tabela estão ordenadas logicamente pelo valor na primeira coluna, que indica o carimbo de tempo (*timestamp*) de chegada (quanto menor o valor, mais antiga a operação).

Saída:
A saída deve ser 1, se um dado escalonamento for estrito, e 0, caso contrário.

Exemplos

Exemplo 01 (escalonamento estrito)

time	#t	op	attr
1	1	R	X
2	2	R	Z
3	1	R	Z
4	3	R	X
5	3	R	Y
6	1	W	X
7	1	C	-
8	3	W	Y
9	3	C	-
10	2	R	Y
11	2	W	Z
12	2	W	Y
13	2	C	-

Exemplo 02 (escalonamento não estrito)

time	#t	op	attr
15	1	R	X
17	2	R	Z
22	1	R	Z
23	3	R	X
26	3	R	Y
34	1	W	X
35	3	W	Y
37	2	R	Y
38	2	W	Z
40	2	W	Y
41	1	C	-
44	2	C	-
48	3	C	-

Saída Exemplo 01	Saída Exemplo 02
`1`	`0`

Submissão

Cada grupo deve entregar um único arquivo *.sql* contendo o algoritmo, instruções básicas para execução, código comentado e testes realizados. A submissão deve ser realizada pelo Google Classroom (o professor criará uma atividade especificamente para isso).

Prazos

Os prazos de entrega do trabalho podem ser consultados no calendário da página do curso no Google Classroom.

Avaliação

O trabalho será avaliado com base nos seguinte critérios:

- Funciona?
- Cumpre os requisitos?
- Qualidade da solução proposta (modelagem, desempenho, etc.)
- Boas práticas (estrutura e organização do código, documentação, etc.)
- Testes

Observações finais

Caso haja algum erro neste documento, serão publicadas novas versões e divulgadas erratas nas aulas. É responsabilidade do(a) aluno(a) manter-se informado(a), frequentando as aulas ou acompanhando as novidades através das ferramentas utilizadas na disciplina.