Universidade Federal do Espírito Santo
Centro Tecnológico
Departamento de Informática

Banco de Dados
Prof. Rodrigo Laiola Guimarães

Instruções gerais: O BOCA é um sistema de correção automática de exercícios que verifica se o resultado gerado pelo seu programa satisfaz casos de teste pré-definidos. Portanto, é necessário seguir estritamente os formatos especificados na questão. Lembre-se que os exemplos dados servem para facilitar o entendimento e podem não cobrir todos os casos de teste que serão usados.

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 01, T_2 lê o valor de *Y* (*time=10*) depois que T_3 o modifica e é confirmada (*time=8* e *time=9*, respectivamente). Já no Exemplo 02, T_2 lê *Y* (*time=37*) antes que T_3, última transação a modificar o item (*time=35*), tenha sido confirmada (*time=48*).

Escreva uma consulta que verifica se um *schedule* é ou não escrito.

Entrada:
Considere a existência da tabela **`Schedule`**, na qual cada linha representa a chegada de uma operação pertencente a uma dada transação (o número de transações presentes no *schedule* pode variar). A tabela possui 4 colunas: a primeira representa o tempo de chegada (`time`), a segunda o identificador da transação (`#t`), a terceira a operação (`read_lock` : bloqueio compartilhado para leitura de um item, `write_lock` : bloqueio (exclusivo) para escrita/gravação de um item, `unlock` : desbloqueio de um item, `read_item` : leitura de um item, `write_item` : escrita de um item, `commit` : confirmação ou `rollback` : aborto/*rollback*) e a quarta o item de dados (atributo) que será bloqueado/desbloqueado/lido/escrito (quando aplicável). As linhas da tabela estão ordenadas logicamente pelo valor na primeira coluna, que indica o carimbo (rótulo) de tempo (*timestamp*) de chegada (quanto menor o valor, mais antiga a operação).

Saída:
A saída deve ser uma tabela contendo uma coluna chamada `RESP` com o valor `1`, se o *schedule* for estrito; caso contrário, `0`.

Exemplo 01

time	#t	op	attr
1	1	read_item	X
2	2	read_item	Z
3	1	read_item	Z
4	3	read_item	X
5	3	read_item	Y
6	1	write_item	X
7	1	commit	-
8	3	write_item	Y
9	3	commit	-
10	2	read_item	Y
11	2	write_item	Z
12	2	write_item	Y
13	2	commit	-

Saída

```
1
```

Exemplo 02

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

Saída

```
0
```