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* **serial**, transações inteiras são realizadas em ordem serial, ou seja, as operações de cada transação são executadas consecutivamente, sem quaisquer operações intercaladas de outra transação; caso contrário, o *schedule* é chamado de **não serial**. Portanto, em um *schedule* serial, somente uma transação de cada vez está ativa – o `commit` (ou `rollback`) da transação ativa inicia a execução da próxima transação. Não ocorre nenhuma intercalação em um *schedule* serial. Uma suposição razoável que podemos fazer, se considerarmos que as transações são *independentes*, é que cada *schedule* serial é considerado *correto*. Podemos assumir isso porque cada transação é considerada correta se executada por conta própria (de acordo com a propriedade de *preservação de consistência*).

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

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 serial; caso contrário, `0`.

Exemplo 01

time	#t	op	attr
1	1	read_item	X
2	1	write_item	X
3	1	read_item	Y
4	1	write_item	Y
5	1	commit	-
6	2	read_item	X
7	2	write_item	X
8	2	rollback	-

Saída

RESP
1

Exemplo 02

time	#t	op	attr
15	2	read_item	X
17	2	write_item	X
22	2	rollback	-
23	1	read_item	X
26	1	write_item	X
34	1	read_item	Y
35	1	write_item	Y
37	1	commit	-

Saída

RESP
1

Exemplo 03

time	#t	op	attr
41	1	read_item	X
52	2	read_item	X
53	1	write_item	X
74	1	read_item	Y
85	2	write_item	X
96	1	write_item	Y
97	1	commit	-
128	2	rollback	-

Saída

```
0
```

Exemplo 04

time	#t	op	attr
135	1	read_item	X
173	1	write_item	X
221	2	read_item	X
223	2	write_item	X
246	2	rollback	-
344	1	read_item	Y
350	1	write_item	Y
372	1	commit	-

Saída

```
0
```