Explain Analyse das operações 0

Na função create\_order\_with\_locks()

EXPLAIN ANALYSE

**SELECT quan\_in\_stock**

**FROM inventory**

**WHERE prod\_id = product\_id;**

**teve de se substituir product\_id por um valor concreto (escolheu-se 9999)**

Segundo o EXPLAIN ANALYSE, a query faz um *Index Scan using inventory\_pkey on inventory*.

Tem-se os seguintes tempos, dados pela média de 10 valores, e incerteza dada pelo desvio padrão:

Planning time = 0,0847 ± 0,001766981 ms

Execution time = 0,045 ± 0,012579 ms

Conclui-se que ????

Na função create\_order\_with\_locks()

EXPLAIN ANALYSE

**SELECT products.price INTO prod\_price**

**FROM products**

**WHERE products.prod\_id = product\_id;**

**teve de se substituir product\_id por um valor concreto (escolheu-se 9999)**

Segundo o EXPLAIN ANALYSE, a query faz um *Index Scan using products\_pkey on products*.

Tem-se os seguintes tempos, dados pela média de 10 valores, e incerteza dada pelo desvio padrão:

Planning time = 0,097 ± 0,004136558 ms

Execution time = 0,0424 ± 0.004033196 ms

Conclui-se que ????

Na função generate\_empty\_order\_with\_locks()

EXPLAIN ANALYSE **SELECT MAX(orderid)+1 FROM orders;**

Segundo o EXPLAIN ANALYSE, a query faz um *Index Only Scan Backward using orders\_pkey on orders*

Tem-se os seguintes tempos, dados pela média de 10 valores, e incerteza dada pelo desvio padrão:

Planning time = 0,1542 ± 0,047760514 ms

Execution time = 0,0873 ± 0,060886141 ms

Conclui-se que ????

Na função createorderline()

EXPLAIN ANALYSE SELECT orderdate FROM orders WHERE orderid = order\_id

**teve de se substituir order\_id por um valor concreto (escolheu-se 12000)**

Segundo o EXPLAIN ANALYSE, a query faz um *Index Scan using orders\_pkey on orders*

Tem-se os seguintes tempos, dados pela média de 10 valores, e incerteza dada pelo desvio padrão:

Planning time =0.0913± 0.010488618 ms

Execution time = 0.0376 ± 0.001349897 ms

Conclui-se que ????

Na função createorderline()

EXPLAIN ANALYSE SELECT MAX(orderlineid) FROM orderlines WHERE orderdate = cdate

**teve de se substituir cdate por um valor concreto (escolheu-se ‘2004-01-01’)**

Segundo o EXPLAIN ANALYSE, a query faz um *Seq Scan on orderlines*

Tem-se os seguintes tempos, dados pela média de 10 valores, e incerteza dada pelo desvio padrão:

Planning time = 0,1598 ± 0,012856473 ms

Execution time = 11,2401 ± 0,968752399 ms

Este query pode ser optimizado criando um índice na chave primária da tabela orderlines. Esta chave consiste nos atributos orderlineid e orderdate:

CREATE INDEX ix\_orderlines\_pkey ON orderlines USING btree(orderlineid, orderdate);

Com esta alteração, fazendo novamente a media de 10 valores, obtém-se os seguintes valores

Planning time = 0,1660 ± 0,008685ms

Execution time = 0, 1306 ± 0,002951 ms

Ou seja, conseguiu-se uma redução para cerca de 1% do tempo de execução.

Nota: como é um índice composto, o pgadmin não permite hash