[12pt, a4paper, twoside, titlepage]article [brazilian]babel [utf8]inputenc [T1]fontenc import [pdftex]graphicx fancyhdr listings color

dkgreenrgb0,0.6,0 grayrgb0.5,0.5,0.5 mauvergb0.58,0,0.82

frame=tb, language=Python, aboveskip=3mm, belowskip=3mm, showstringspaces=false, columns=flexible, basicstyle=, numbers=none, numberstyle=gray, keywordstyle=blue, commentstyle=dkgreen, stringstyle=mauve, breaklines=true, breakatwhitespace=true, tabsize=3

fancyplain [L] [C] [R] 0pt 0pt 13.6pt

[1]1

Universidade Estadual de Santa Cruz

Eduardo Reis Nobre

document

Introduo Durante a implementao de um banco de dados, devemos considerar que tipo de aplicao teremos e se a mquina escolhida para o servio capaz de atender a aplicao. A capacidade de simular cenrios e ver como sua mquina reagir a tais cenrios pode ser obtida de duas formas: atravs de uso real, onde se utilizar uma configurao mdia sendo armazenado as informaes relevantes ou atravs de uma ferramenta de benchmark que capaz de realizar diversos testes em diversos cenrios sem a necessidade da interao com usurio lhe dando o controle sobre diversos aspectos da simulao. Esse relatrio fala do uso do pgbench para a o teste de performance do banco de dados postgres. Ambiente Os testes foram realizados utilizando um computador com a seguinte configurao:

Os testes foram realizados utilizando-se o sgbd postgres verso 9.5.

Metodologia documentclasses Todos os cenrios foram testados 10 vezes, sendo obtidas ento as mdias de Latncia e quantidade de transaes por segundo. As chamadas foram realizadas para um banco com um fator de escala de 10 utilizando um script em python3 para a realizao das chamadas, clculo e armazenamento dos resultados para posterior anlise.

def $\mathrm{make}_c sv_b atch(client = 10, transaction = 10) : body ='' lines ='' for index in range(1, BATCH_S IZE+$ $1) : process = subprocess.Popen('pgbench-d-C-c-tbenchmark'.format(client, transaction).split(), stdout =$ $subprocess.PIPE)output, error = process.communicate()line = output.decode().split('')[8 : 12]lines =$ $lines + \mathrm{make}_c sv_l ine(line)$

body = body + $\mathrm{make}_m ean(lines)with open(OUT_P ATH+'out_t 5x.csv'.format(client, transaction),' w')as outfile :$ $outfile.write(body)$

def make(): for client in CLIENTS: for transaction in TRANSACTIONS: $\mathrm{make}_c sv_b atch(client, transaction)$