**kdb+性能测试**

**笔者：**陈泽天

**日期：**2019-07-23

­横向速度测试

**目的**

测试kdb+在行数相同的情况下，当列数增多的时候，csv文件导入，hdb（历史数据库）导出，hdb导入，平均值计算用时所产生的变化。

**操作**

每次生成一张1，000，000行的表格，列数分别为10+10列、10+20列、10+50列、10+100列。前10列分别为日期（date）、小时（hour）、分钟（minute）、股票代码（ticker）、开盘价（op）、最高价（hp）、最低价（lp）、收盘价（cp）、股数（volume）、总额（amount）。这些数据都将尽可能根据实际情况随机生成。其余每列数据为从10000～1000000之间的随机浮点数。每次计算每行随机数的平均数，并记录各项用时。

**具体过程**

首先，编写python脚本去生成csv文件。脚本如下：

import random

if \_\_name\_\_ == "\_\_main\_\_":

# Number of rows that are totally random

rand\_rows\_num = 10

# Number of records in the table

records\_num = 10

with open('fakedata.csv', 'w') as f:

# Write the column names

f.write("date,hour,minute,ticker,op,hp,lp,cp,volume,amount")

for row in range(0, rand\_rows\_num):

f.write(",random{}".format(row))

f.write("\n")

# Write the data

for \_ in range(records\_num):

# Randomly if the transaction happens in the morning or afternoon

morning = random.randint(0, 1)

if morning == 0: # Afternoon, time period = [13:01, 15:00]

hour = random.randint(13, 15)

if hour == 13:

minute = random.randint(1, 59)

elif hour == 14:

minute = random.randint(0, 59)

else:

minute = 0

else: # Morning, time period = [9:31, 11:30]

hour = random.randint(9, 11)

if hour == 9:

minute = random.randint(31, 59)

elif hour == 10:

minute = random.randint(0, 59)

else:

minute = random.randint(0, 30)

# ticker is a 6-digit code

# from Shenzheng (SZ), Shanghai (SH), Hong Kong (HK)

ticker\_number = random.randint(0, 999999)

ticker\_index = random.randint(0, 2)

ticker = "{:06d}.{}".format(ticker\_number, ["SZ", "SH", "HK"][ticker\_index])

# Let lowest price and highest price be within [10, 90]

prices = [random.uniform(10, 90) for \_ in range(2)]

hp = max(prices)

lp = min(prices)

# Let opening price and closing price be within the lowest price and highest price

op = random.uniform(lp, hp)

cp = random.uniform(lp, hp)

# Let volume be within [1000000, 10000000]

volume = random.randint(1000000, 10000000)

# Let amount be amount = volume \* some random number between the lowest price and highest price

amount = volume \* random.uniform(lp, hp)

# Write the values into the csv file

f.write("20190603,{},{},{},{},{},{},{},{},{}".format(hour, minute, ticker, op, hp, lp, cp, volume, amount))

# Randomly generate floats

for \_ in range(rand\_rows\_num):

value = random.uniform(10000, 1000000)

f.write(", {}".format(value))

f.write("\n")

f.close()

print("Done.")

末尾