**孝庸数据压缩存储方案**

**问题描述**

我们现在每天接收和处理的数据占用大量的存储空间，仅考虑常规的股票和期货数据（CTP和宏汇），每日产生超过30G的数据，约6万个文件。

原始数据：

CTP 4G

宏汇 7G （已压缩）

资金值 8M

衍生数据：

MktInfo 2M

Index 200M

Market 6G

Trans 8G

Order 5G

Halfcook 200M

Tick 2G

K线 200M

由此带来的问题至少有3个：

* 服务器存储空间消耗大。粗略估计现在每年需要新增存储空间7.5T。未来还可能更多。
* 磁盘I/O成为数据处理和应用的瓶颈，限制了数据吞吐能力。
* 海量的小文件进一步拉低了存储效率和读写性能。

**解决方案**

* 原始数据保持现有格式不变，定期归档。归档后原服务器上的数据可以删除。
* 衍生数据采用压缩存储，压缩格式为7zip。每个数据类型中的每个交易代码，以月为单位压缩为单一文档。例如2019年10月10600000（浦发银行）的所有Tick数据压缩为一个文档。
* 支持分布式存储，每一个压缩后的文档可以存放在任何一台服务器的任何一个磁盘上。
* 提供标准数据接口，支持C++和Python。支持Windows 10和CenOS 7操作系统。
* 采用自动化工具每日对数据进行压缩，存放在预定位置。
* 支持7x24小时并发访问。数据处理期间不影响数据的正常使用。

**存储结构**

压缩文档 : <HOME>\<TYPE>\<MONTH>\<CODE>.7z

数据文件 : <DATE>.dat

例如

<HOME> : [\\192.168.2.3\XYData](file://\\192.168.2.3\XYData)

<TYPE> : Tick

<MONTH> : 201910

<CODE> : 10600000

<DATE> : 20191008

则

压缩文档 : [\\192.168.2.3\XYData\Tick\201910\10600000.7z](file://\\192.168.2.3\XYData\Tick\201910\10600000.7z)

其中包含的数据文件为: 20191008.dat

**访问接口**

**C++接口**

class XYData

{

public:

// 静态函数，用于创建一个对象

// 参数

// path – 数据根目录，例如: “\\\\192.168.2.3\\XYData”

static XYData \* New(const std::string & path);

// 销毁一个对象

virtual void Release();

// 读取数据并解压

// 参数

// type – 数据类型。当前可用的数据类型包括："Tick", "K60", "MktInfo",

// "Halfcook", "Index", "Market", "Trans", "Order"

// date – 数据日期，例如 20191008

// code – 交易代码，例如 "10600000" 或 "600000.SH", "20000001" 或

// "000001.SZ", "IF888"

// output – 若数据读取成功，返回已解压的数据

// 返回值

// EC\_OK – 数据读取成功

// EC\_DATA – 数据不存在

// EC\_FATAL – 严重错误

virtual XYErrorCode Read(

const std::string & type,

const std::string & code,

int date,

std::vector<unsigned char> & output);

// 查询数据

// 参数

// type – 数据类型。当前可用的数据类型包括："Tick", "K60", "MktInfo",

// "Halfcook", "Index", "Market", "Trans", "Order"

// date – 数据日期，例如 20191008

// code – 交易代码，例如 "10600000" 或 "600000.SH", "20000001" 或

// "000001.SZ", "IF888"

// 返回值

// EC\_OK – 查询数据成功

// EC\_DATA – 数据不存在

// EC\_FATAL – 严重错误

virtual XYErrorCode Query(

const std::string & type,

const std::string & code,

int date);

};

**C++代码示例**

#include <iostream>

#include "XYData.h"

#ifdef \_WIN32

#define HOME "\\\\shanghai\\XYData"

#define DLL7Z "7z.dll"

#else

#define HOME "/home/oliver/XYData"

#define DLL7Z "7z.so"

#endif

int main()

{

// Create data reader

XYData \* pData = XYData::New(HOME);

if (!pData) {

std::cout << "Can not find " DLL7Z << std::endl;

return -1;

}

// Query data

XYErrorCode error = pData->Query("Tick", "000001.SZ", 20191008);

if (error == EC\_OK)

std::cout << "Data exists." << std::endl;

else if (error == EC\_DATA)

std::cout << "Data does not exist." << std::endl;

else

std::cout << "Fatal error. Please report to system admin.";

error = pData->Query("Market", "600000.SH", 20200610);

if (error == EC\_OK)

std::cout << "Data exists." << std::endl;

else if (error == EC\_DATA)

std::cout << "Data does not exist." << std::endl;

else

std::cout << "Fatal error. Please report to system admin.";

// Read data

std::vector<unsigned char> buffer;

error = pData->Read("Tick", "000001.SZ", 20191008, buffer);

if (error == EC\_OK)

std::cout << "Data size = " << buffer.size() << std::endl;

else if (error == EC\_DATA)

std::cout << "Data does not exist." << std::endl;

else

std::cout << "Fatal error. Please report to system admin.";

// Clear the buffer bebore reading

buffer.clear();

error = pData->Read("Market", "600000.SH", 20200610, buffer);

if (error == EC\_OK)

std::cout << "Data size = " << buffer.size() << std::endl;

else if (error == EC\_DATA)

std::cout << "Data does not exist." << std::endl;

else

std::cout << "Fatal error. Please report to system admin.";

// Release data reader

pData->Release();

return 0;

}

**Python代码示例**

import platform

from XYData import XYData

if platform.system() == "Linux":

data = XYData("/home/oliver/XYData")

else:

data = XYData("\\\\shanghai\\XYData")

# 简单用法

result = data.Read("Tick", "000001.SZ", 20191008)

if result:

print("Data size = ", len(result))

print("Data = ", result[0:min(100, len(result))])

else:

print("Data does not exist.")

# 高级用法

size = data.Load("Market", "600000.SH", 20200610)

if size > 0:

print("Data size = ", size)

print("Data = ", data.Get(0, min(100,size)))

else:

print("Data does not exist.")

# 查询数据

print(data.Query("Tick", "000001.SZ", 20191008))

print(data.Query("Market", "600000.SH", 20200610))