## **vn.py社区精选22 - 看完这篇，彻底学会CSV历史数据导入！**

原创 用Python的交易员 VNPY  2020-05-11

收录于话题

#vn.py社区精选

25个

**原文作者：李怡然**

如何将本地CSV文件格式的历史数据导入到数据库中，是vn.py社区论坛上提问最多的问题之一。本文的主要目的是帮助初学者解决数据入库问题，以便可以快速开始量化策略的开发研究。

本文的内容主要分为三大部分：

第一部分介绍了在vn.py中使用MongoDB数据库所需要进行的配置（只打算使用vn.py默认SQLite数据库的用户，可以简单了解一下）；

第二部分介绍了数据入库的基本流程（适用于vn.py支持的所有数据库）；

最后一部分则是具体的实现：分别将数据导入MongoDB和SQLite数据库（适用于vn.py支持的所有数据库）。

在正文开始之前，还需要提醒大家：将数据导入数据库之前，务必要确保这些数据已经是被清洗过的干净的数据。如果将没有被清洗过质量差的数据，直接入库进行回测，可能会导致各种问题，影响策略开发的效率。因此，建议大家使用高质量的数据源。

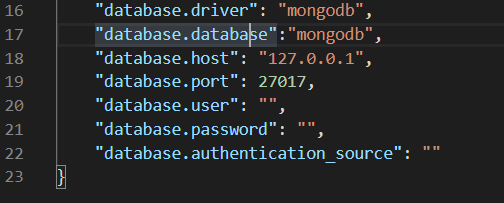
#### ****配置数据库****

[vn.py](http://vn.py/" \o "http://vn.py) 中默认使用 SQLite 数据库。因此，如果需要使用 MongoDB 数据库，则需要修改 [vn.py](http://vn.py/" \o "http://vn.py) 的全局配置。具体流程如下：

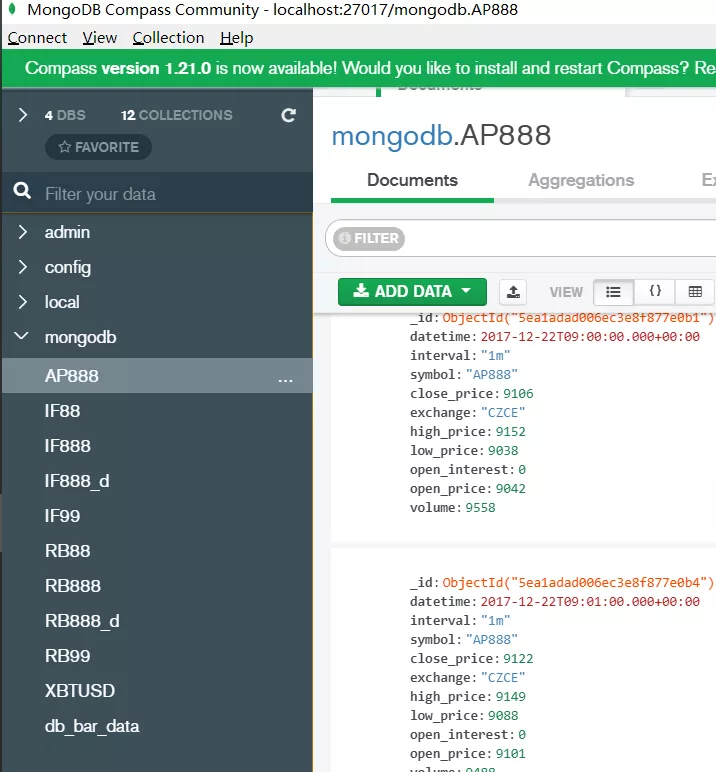
找到 vt\_setting.json 文件：位于C:\Users\你的用户名\.vntrader 目录下；

对vt\_setting.json 文件中的相关字段进行修改，包括：database.driver、database.database、database.host、database.port。

下图是我自己设置的数据库配置信息：



上图中的两个 "mongodb" 可能让人会有些困扰。实际上第一个"mongodb"是告诉 [vn.py](http://vn.py/" \o "http://vn.py) 我们使用的数据库类型是 MongoDB 数据库而不是默认的 SQLite 数据库。第二个 "mongodb" 则是告诉 [vn.py](http://vn.py/" \o "http://vn.py) 回测所需要的数据储存在 MongoDB 数据库中一个叫做 "mongodb" 的 database 中。这样说可能有些绕口，请看下图:



上图是 MongoDB 数据库的官方图形界面客户端MongoDB Compass。我们可以清楚的看到在该 MongoDB 数据库中一共有四个 database， 分别是 admin，config，local, mongodb。从2.0开始，vn.py采用ORM/ODM的方式来管理历史数据，因此所有的K线数据都被保存在db\_bar\_data这一集合中，所有Tick数据都被保存在db\_tick\_data这一集合中。

#### ****数据入库基本流程****

在配置好 MongoDB 数据库后，我们可以正式开始讨论数据入库操作，[vn.py](http://vn.py/" \o "http://vn.py) 提供了很多工具使数据入库这个过程变得简单快捷，以下是是数据入库的基本流程：

先确定要入库的数据是 Tick 还是 Bar (K线) 类型数据；

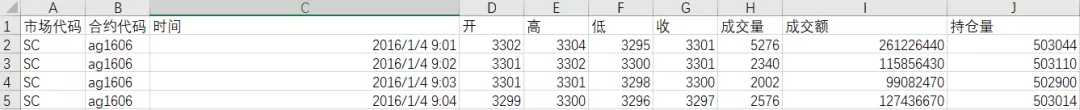
将需要入库的数据转化成 [vn.py](http://vn.py/" \o "http://vn.py) 定义的 TickData 或 BarData 数据类型；

使用 [vn.py](http://vn.py/" \o "http://vn.py) 提供的数据入库工具函数 database\_manager.save\_tick\_data 或 database\_manager.save\_bar\_data 将相应的 TickData 或 BarData 入库  。

以上三步操作中的难点（或者说复杂点）集中在第二步，即如何将将本地CSV文件中的数据格式转换成vn.py 定义的 TickData 或 BarData，大部分情况下只需对数据的时间戳进行转化处理。另外，如果数据本身的质量不高，比如数据的整个时间戳格式前后不一致，那需要先对数据的时间戳格式进行统一。

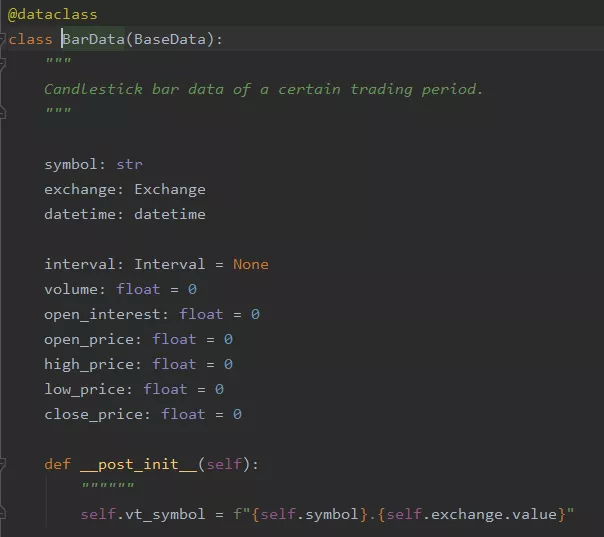
#### ****数据入库具体实现****

在了解了数据入库的基本流程之后，我们来实现一次数据入库的过程。首先，来看一看我们要入库的数据：



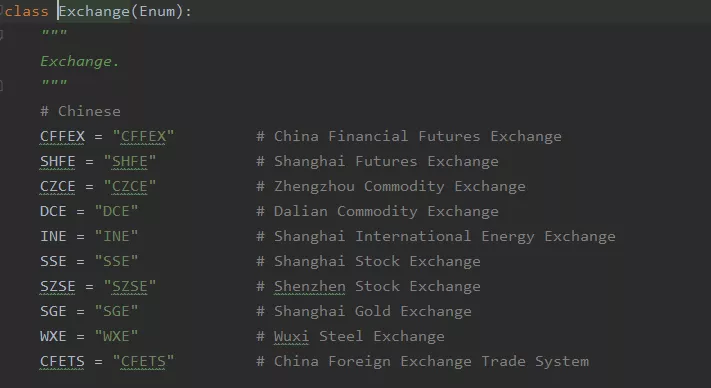
从上图可以看出，C 列储存的是表示时间的数据且 C2 和 C3 的时间间隔是1分钟。所以，要入库的数据是1分钟的 Bar (K线)数据类型。下面我们进行第二步：将需要入库的数据转化成 [vn.py](http://vn.py/" \o "http://vn.py) 定义的 BarData。

首先，我们先来认识一下 [vn.py](http://vn.py/" \o "http://vn.py) 中的BarData：

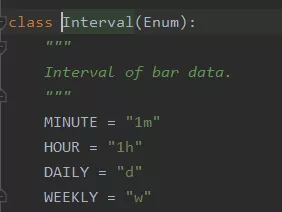


从上图中可以看出，BarData 一共有11个属性。其中，BarData.vt\_symbol 会在 BarData 实例化的时候自动生成。另外，需要指出的是BarData.exchange 和 BarData.inteval 的数据类型分别是 [vn.py](http://vn.py/" \o "http://vn.py) 中定义好的枚举常量 Exchange 和 Inteval 而 BarData.datetime 则是 Python 标准库 datetime 中的 datetime 数据类型。

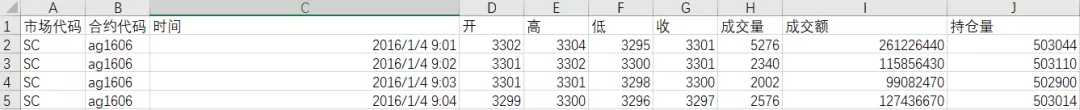
Exchange 枚举值的定义：



Interval 枚举值的定义：



在认识了vn.py 中的 BarData 之后，我们开始着手将需要入库的数据转化成 BarData类型数据。再来重温一下，需要入库数据的格式：



通过和上文 BarData 的数据结构对比，我们有以下几个发现：

csv文件中的 合约代码 时间 开 高 低 收 成交量 持仓量 和 BarData中的 symbol datetime open\_price high\_price low\_price close\_price volume open\_interest 一一对应(从名称就就可以看出)。

csv文件中 市场代码 没办法和 BarData 中的 exchange 对应。因为csv文件中 市场代码 都是 SC ，而在上图Exchange 数据结构代码截图中找不到和 SC 对应的枚举常量的绑定值。从合约代码 ag1608(沪银1608) 可以推断出这里的 SC 指的就是上海期货交易所，对应的枚举常量是 Exchang.SHFE。

csv文件中缺少了和 BarData 中的 interval 相对应的数据。上文我们已经发现了 csv文件中储存的是1分钟的BarData，对应的枚举常量是 Interval.MINUTE。

基于上面的发现，很自然的，我们需要进行如下的操作：

将csv文件中 市场代码的 SC 替换成 Exchang.SHFE

增加一列数据，且该列数据的所有值都是 Interval.MINUTE

一般情况下，使用Python 的 pandas 库可以方便的完成上面的操作。如果数据的质量较差，比如数据的分隔符设置存在问题，会使得pd.read\_csv函数没办法正确的读取.csv文件。这时则需要使用Python的 csv库。本文的数据入库过程统一使用 pandas 来完成。具体操作，如下：

from vnpy.trader.constant import (Exchange, Interval)import pandas as pd*# 读取需要入库的csv文件，该文件是用gbk编码*imported\_data = pd.read\_csv('需要入库的数据的绝对路径',encoding='gbk')  
*# 将csv文件中 `市场代码`的 SC 替换成 Exchange.SHFE SHFE*imported\_data['市场代码'] = Exchange.SHFE  
*# 增加一列数据 `inteval`，且该列数据的所有值都是 Interval.MINUTE*imported\_data['interval'] = Interval.MINUTE

接下来，我们还需要对每列数据的数据类型进行修改，确保和 BarData 中各个属性的数据类型一致。BarData中属性的数据类型可以分为三大类：float 类， datetime 类 和 自定义枚举类 (Interval 和 Exchange)。因为，上面已经修改过了Interval 和 Exchange，下面只需要修改 float 和 datetime 类。

修改 float 类代码：

*# 明确需要是float数据类型的列*float\_columns = ['开', '高', '低', '收', '成交量', '持仓量']  
for col in float\_columns: imported\_data[col] = imported\_data[col].astype('float')

修改 datatime 类代码：

*# 明确时间戳的格式# %Y/%m/%d %H:%M:%S 代表着你的csv数据中的时间戳必须是 2020/05/01 08:32:30 格式*datetime\_format = '%Y%m%d %H:%M:%S'  
imported\_data['时间'] = pd.to\_datetime(imported\_data['时间'],format=datetime\_format)

下一步，我们还需要对列名进行修改：

*# 因为没有用到 成交额 这一列的数据，所以该列列名不变*imported\_data.columns = ['exchange','symbol','datetime','open','high','low','close','volume','成交额','open\_interest','interval']

另外，因为该csv文件储存的是ag的主力连续数据，即多张ag合约的拼接。因此，symbol列中有多个不同到期日的ag合约代码，这里需要将合约代码统一为ag88：

imported\_data['symbol'] ='ag88'

最后，我们使用 vn.py 封装好的 database\_manager.save\_bar\_data 将数据入库：

*# 导入 database\_manager 模块*from vnpy.trader.database import database\_managerfrom vnpy.trader.object import (BarData,TickData)*# 封装函数*def move\_df\_to\_mongodb(imported\_data:pd.DataFrame,collection\_name:str): bars = [] start = None count = 0  
 for row in imported\_data.itertuples():  
 bar = BarData(  
 symbol=row.symbol, exchange=row.exchange, datetime=row.datetime, interval=row.interval, volume=row.volume, open\_price=row.open, high\_price=row.high, low\_price=row.low, close\_price=row.close, open\_interest=row.open\_interest, gateway\_name="DB",  
 )  
  
 bars.append(bar)  
 *# do some statistics* count += 1 if not start: start = bar.datetime end = bar.datetime  
 *# insert into database* database\_manager.save\_bar\_data(bars)    print(f"Insert Bar: {count} from {start} - {end}")

如果想要将数据储存储存在 SQLite 数据库中也很简单，只需要两步就可以完成。

首先创建一个sqlite数据库连接对象：

from vnpy.trader.database.initialize import init\_sqlfrom vnpy.trader.database.database import Driver  
settings={ "database": "database.db", "host": "localhost", "port": 3306, "user": "root", "password": "", "authentication\_source": "admin"}sqlite\_manager = init\_sql(driver=Driver.SQLITE, settings=settings)

然后使用sqlite数据库连接对象将数据入库：

*# 替换函数 move\_df\_to\_mongodb 的倒数第二行*sqlite\_manager.save\_bar\_data(bars)

#### 如果在进行Sqlite数据入库的时候，出现peewee.InterfaceError: Error binding parameter 2 - probably unsupported type错误，解决方法如下：

找到imported\_data['时间'] = pd.to\_datetime(imported\_data['时间'],format=datetime\_format)代码所在行

在该行代码下键入imported\_data['时间'] = imported\_data['时间'].dt.strftime('%Y%m%d %H:%M:%S')

详细的Debug过程记录在[sqlite数据入库Debug](https://github.com/Yiran13/Strategies/tree/master/debug_process" \o "https://github.com/Yiran13/Strategies/tree/master/debug_process). 将该文件夹内的内容下载到本地同一个位置，运行Jupyter Notebook就可以复现整个过程.

#### ****总结****

本文尝试从数据库配置，数据入库基本流程，数据入库具体实现，三部分来帮助vn.py新用户解决编写python脚本实现数据入库这个难点。借助vn.py的database\_manager模块，用户基本上可以无缝切换SQLite，MongoDB等vn.py支持的数据库来读取和存入数据。希望这篇文章能帮助大家快速进入量化策略的研究和开发。

#### ****最后附上完整代码****

from vnpy.trader.constant import (Exchange, Interval)import pandas as pdfrom vnpy.trader.database import database\_managerfrom vnpy.trader.object import (BarData,TickData)  
*# 封装函数*def move\_df\_to\_mongodb(imported\_data:pd.DataFrame,collection\_name:str): bars = [] start = None count = 0  
 for row in imported\_data.itertuples():  
        bar = BarData( symbol=row.symbol, exchange=row.exchange, datetime=row.datetime, interval=row.interval, volume=row.volume, open\_price=row.open, high\_price=row.high, low\_price=row.low, close\_price=row.close, open\_interest=row.open\_interest,              gateway\_name="DB",        ) bars.append(bar)  
 *# do some statistics* count += 1 if not start: start = bar.datetime end = bar.datetime  
 *# insert into database* database\_manager.save\_bar\_data(bars, collection\_name) print(f'Insert Bar: {count} from {start} - {end}')  
if \_\_name\_\_ == "\_\_main\_\_": *# 读取需要入库的csv文件，该文件是用gbk编码* imported\_data = pd.read\_csv('D:/1分钟数据压缩包/FutAC\_Min1\_Std\_2016/ag主力连续.csv',encoding='gbk') *# 将csv文件中 `市场代码`的 SC 替换成 Exchange.SHFE SHFE* imported\_data['市场代码'] = Exchange.SHFE *# 增加一列数据 `inteval`，且该列数据的所有值都是 Interval.MINUTE* imported\_data['interval'] = Interval.MINUTE *# 明确需要是float数据类型的列* float\_columns = ['开', '高', '低', '收', '成交量', '持仓量'] for col in float\_columns: imported\_data[col] = imported\_data[col].astype('float') *# 明确时间戳的格式* *# %Y/%m/%d %H:%M:%S 代表着你的csv数据中的时间戳必须是 2020/05/01 08:32:30 格式* datetime\_format = '%Y%m%d %H:%M:%S' imported\_data['时间'] = pd.to\_datetime(imported\_data['时间'],format=datetime\_format) *# 因为没有用到 成交额 这一列的数据，所以该列列名不变* imported\_data.columns = ['exchange','symbol','datetime','open','high','low','close','volume','成交额','open\_interest','interval'] imported\_data['symbol'] ='ag88'    move\_df\_to\_mongodb(imported\_data,'ag88')