mongo2file

发布 *v1*

PY-GZKY

2022 年 03 月 08 日

Contents

↻ ***Mongo2file*** 是一个 Mongodb 数据库转换为表格文件的库。

CHAPTER 1

快速

mongo2file 依赖于 PyArrow 库。它是 C++ Arrow 的 Python 版本实现。

PyArrow 目前与 Python 3.7、3.8、3.9 和 3.10 兼容。

如果您在 Windows 上遇到任何的导入问题或错误，您可能需要安装 Visual Studio 2015。

警告:

PyArrow 目前只支持到 win64 位 (Python 64bit) 操作系统。

其次, 除了常见的 csv、excel、以及 json 文件格式之外, mongo2file 还支持导出 pickle、feather、parquet 的二进制压缩文件。

pickle、feather、parquet 是 Python 序列化数据的一种文件格式, 它把数据转成二进制进行存储。从而大大减少的读取的时间。

CHAPTER 2

依赖

- PyArrow >= 7.0.0

2.1 安装

```
pip install mongo2file
```

CHAPTER 3

基本用法

3.1 快速开始

```
import os
from mongo2file import MongoEngine

M = MongoEngine(
    host=os.getenv('MONGO_HOST', '127.0.0.1'),
    port=int(os.getenv('MONGO_PORT', 27017)),
    username=os.getenv('MONGO_USERNAME', None),
    password=os.getenv('MONGO_PASSWORD', None),
    database=os.getenv('MONGO_DATABASE', 'test_'),
    collection=os.getenv('MONGO_COLLECTION', 'test_')
)


def to_csv():
    result_ = M.to_csv()
    assert "successfully" in result_


def to_excel():
    result_ = M.to_excel()
    assert "successfully" in result_
```

(下页继续)

(续上页)

```
def to_json():
    result_ = M.to_excel()
    assert "successfully" in result_


def to_pickle():
    result_ = M.to_pickle()
    assert "successfully" in result_


def to_feather():
    result_ = M.to_feather()
    assert "successfully" in result_


def to_parquet():
    result_ = M.to_parquet()
    assert "successfully" in result_
```

当 MongoEngine 控制类指定了 mongodb 表名称时、将对数据表 (mongodb 集合) 进行导出操作。

其类方法参数包括:

- query: 指定对数据表的查询参数、只对指定表名时有效
- folder_path: 指定导出目录路径
- filename: 指定导出文件名、默认为 表名称 + 当前时间
- _id: 指定是否导出 _id、布尔型、默认为 False
- limit: 指定导出表的限制数据、int 类型、默认为 -1、即不限制。

```
import os
from mongo2file import MongoEngine

"""
作用于 MongoEngine 类未指定表名称时
"""

M = MongoEngine(
    host=os.getenv('MONGO_HOST', '127.0.0.1'),
    port=int(os.getenv('MONGO_PORT', 27017)),
    username=os.getenv('MONGO_USERNAME', None),
    password=os.getenv('MONGO_PASSWORD', None),
```

(下页继续)

(续上页)

```
    database=os.getenv('MONGO_DATABASE', 'test_')
)


def to_csv():
    result_ = M.to_csv()
    assert "successfully" in result_


def to_excel():
    result_ = M.to_excel()
    assert "successfully" in result_


def to_json():
    result_ = M.to_json()
    assert "successfully" in result_
```

当 MongoEngine 控制类只指定了 mongodb 库名称时、将对数据库下所有集合进行导出操作。

CHAPTER 4

mongo2file 的改进

对于 mongodb 的全表查询、条件查询、聚合操作、以及索引操作 (当数据达到一定量级时建议) 并不是直接影响数据导出的最大因素。

因为 mongodb 的查询一般而言都非常快速，主要的瓶颈在于读取数据库之后将数据转换为大列表存入表格文件时所耗费的时间。

这是一件非常可怕的事情。

当没有多线程 (当然这里的多线程并不是对同一文件进行并行操作，文件写入往往是线程不安全的)、数据表查询语句无优化时，并且当数据达到一定量级时 (比如 100w 行)，单表单线程表现出来的效果真是让人窒息。

在 mongo2file 在进行大数据量导出时表现的并没有多么优秀。导致的主要原因可能是:

- 采用的 xlsxwriter 库写入 excel 时是积极加载 (非惰性) 的，数据全部加载至内存后插入表格。
- 大数据量插入表格时、跟宿主机器的性能有关。

mongo2file 表现的不如人意时，我做出了一下改进:

- 当数据量过大时，数据表分块读取，导出多表格。
- 增加线程池的最大并发数、当选取的 block_size 值合适时，将发挥最大性能。

CHAPTER 5

一些 Fix

- 对于 xlsxwriter、openpyxl、xlwings 以及 pandas 引用的任何引擎进行写入操作时、都会对写入数据进行非法字符的过滤。这一点从部分源码中可以看得出来。
- 由于行数据表中可能存在 excel 无法识别的非法字符(比如空列表 []),当写至此行时将抛出 非法类型的错误。
- 而比较恰当合理的做法就是在存储 mongodb 文档时不要存入类似于 []、{} 的这种对原始数据无意义的空对象。

CHAPTER 6

Reference

6.1 *MongoEngine*

```
MongoEngine(
    host='localhost',
    port=27017,
    username=None,
    password=None,
    database='测试库',
    collection='测试表_200000'
)
```

6.1.1 *to_csv(query, folder_path, filename, ...)*

```
:param query: 数据库查询条件、字典类型、只作用于单表导出
:param folder_path: 指定导出的目录
:param filename: 指定导出的文件名
:param _id: 是否导出 _id 默认否
:param limit: 限制数据表查询的条数
:param is_block: 是否分块导出
:param block_size: 块大小、is_block 为 True 时生效
```

6.1.2 *to_excel(query, folder_path, filename, ...)*

```
:param query: 数据库查询条件、字典类型、只作用于单表导出
:param folder_path: 指定导出的目录
:param filename: 指定导出的文件名
:param _id: 是否导出 _id 默认否
:param limit: 限制数据表查询的条数
:param is_block: 是否分块导出
:param block_size: 块大小、is_block 为 True 时生效
:param mode: 导出模式, 枚举类型、sheet 或 xlsx, 当 is_block 为 True 时生效
:param ignore_error:␣
↪是否忽略错误、数据表中存在非序列化类型时使用、这将一定程度上影响程序的性能
```

6.1.3 *to_json(query, folder_path, filename, ...)*

```
:param query: 数据库查询条件、字典类型、只作用于单表导出
:param folder_path: 指定导出的目录
:param filename: 指定导出的文件名
:param _id: 是否导出 _id 默认否
:param limit: 限制数据表查询的条数
:param is_block: 是否分块导出
:param block_size: 块大小、is_block 为 True 时生效
```

6.1.4 *to_pickle(query, folder_path, filename, ...)*

```
:param query: 数据库查询条件、字典类型、只作用于单表导出
:param folder_path: 指定导出的目录
:param filename: 指定导出的文件名
:param _id: 是否导出 _id 默认否
:param limit: 限制数据表查询的条数
```

6.1.5 *to_feather(query, folder_path, filename, ...)*

```
:param query: 数据库查询条件、字典类型、只作用于单表导出
:param folder_path: 指定导出的目录
:param filename: 指定导出的文件名
:param _id: 是否导出 _id 默认否
:param limit: 限制数据表查询的条数
```

6.1.6 *to_parquet(query, folder_path, filename, ...)*

```
:param query: 数据库查询条件、字典类型、只作用于单表导出
:param folder_path: 指定导出的目录
:param filename: 指定导出的文件名
:param _id: 是否导出 _id 默认否
:param limit: 限制数据表查询的条数
```

CHAPTER 7

致谢

- Arrow

CHAPTER 8

License

- Apache License