Crawler - HW1 (Scrapy)

Friday, 18.07.2020

Contributors

Navid Jafarof
Amirali Monjar

Summary

This project is an implementation of a crawler which scrapes stocks data from FipIran, using Python 3.8, Scrapy and SqlAlchemy.

Project Structure

The project mainly consists of 3 parts.

1. A command line interface.
2. A spider that crawls the web pages.
3. A database handler which stores and reads data in and from SqlLite.

Crawling Stock Names

In the fipiran.py file we implement an spider which firstly, sends Get request to 'http://www.fipiran.com/Market/LupBourse' and retrieves the names of the stocks (336 at the time)

```
def parse_stock_names(self, response):
   response_string = response.body.decode("utf-8")
   extractor = Extractor(response_string[response_string.index("<table"):
response_string.index("</table>")])
   extractor.parse()
   names = [row[0] for row in extractor.return_list()]
   for name in names:
       time.sleep(1)
```

```
        stock = Stock(name=name)
        added_id = db_handler.add_stock(stock)
        url = ut.crawl_url_generator(name, manager.rows_num, 1)
        yield scrapy.Request(url=url,
callback=self.make_stock_info_parser(added_id, manager.is_update,
                                                                              name
== names[len(names) - 1]), dont_filter=True)
```

Crawling Stock Daily Information

The second part of the spider is a function that crawls http://www.fipiran.com/Symbol/HistoryPricePaging and passes rows number to get a specific amount of data in each crawl.

```
def make_stock_info_parser(self, stock_id, is_update, is_last):
    def parse(response):
        infos = json.loads(response.body)['data']
        for info in infos:
            info["stock_id"] = stock_id
            info_object = StocksDailyInfo(info)
            db_handler.add_info_for_stock(stock_id, info_object)
        if is_last:
            if is_update:
                print("WAITING...")
                time.sleep(3600 * 24 * 1)
                print("STARTED UPDATE")
                yield scrapy.Request(url=self.first_page_url,
callback=self.parse_stock_names,dont_filter=True)

    return parse
```

Database

The databasehandler.py script handles all database transactions using sqlAlchemy library and ORM:

```
class DataBaseHandler:
    session = None
    def init_database(self):
        engine = create_engine('sqlite:///fipiran.db')
        session_class = sessionmaker(bind=engine)
        self.session = session_class()
        base.metadata.create_all(engine)

    def add_all(self, items):
        self.session.add_all(items)
        self.session.commit()

    def add_stock(self, stock):
        if self.session.query(Stock).filter_by(name=stock.name).scalar() is not
None:
            return
self.session.query(Stock).filter_by(name=stock.name).first().id
        else:
            self.session.add(stock)
            self.session.commit()
            return stock.id

    def add_info_for_stock(self, stock_id, info):
        stock = self.session.query(Stock).get(stock_id)
        if len(list(filter(lambda inf: inf.gDate == info.gDate,
stock.stock_daily_infos))) == 0:
            stock.stock_daily_infos.append(info)
            self.session.commit()

    def get_infos(self):
        return self.session.query(StocksDailyInfo).all()

    def get_stocks(self):
        return self.session.query(Stock).all()

    def get_database_json(self):
        stock_list = []
        for stock in self.get_stocks():
            stock_list.append(stock)
        data = {"stocks": stock_list}
        json_string = json.dumps(data, cls=build_alchemy_encoder(),
check_circular=False, ensure_ascii=False).encode(
            'utf8').decode()
        return json_string
```